(A3-1)多媒體遊戲設計實務

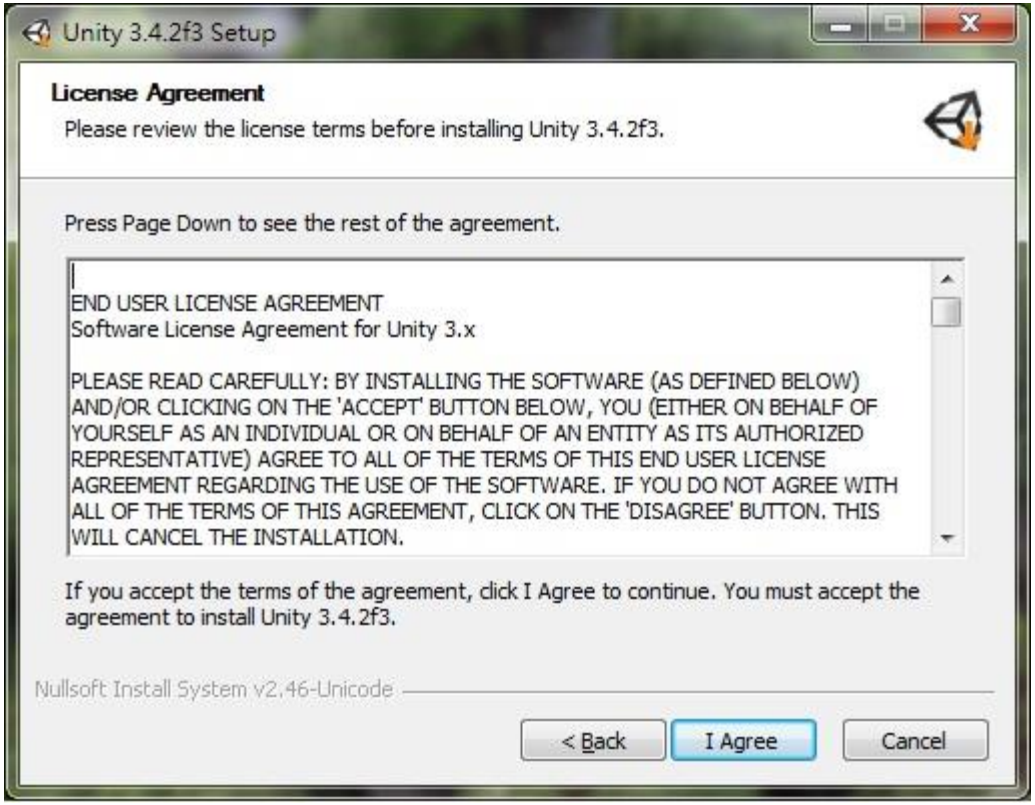
德霖技術學院
資訊工程系
105學年度第2學期
授課教師 :臧意周

單元一 開發環境

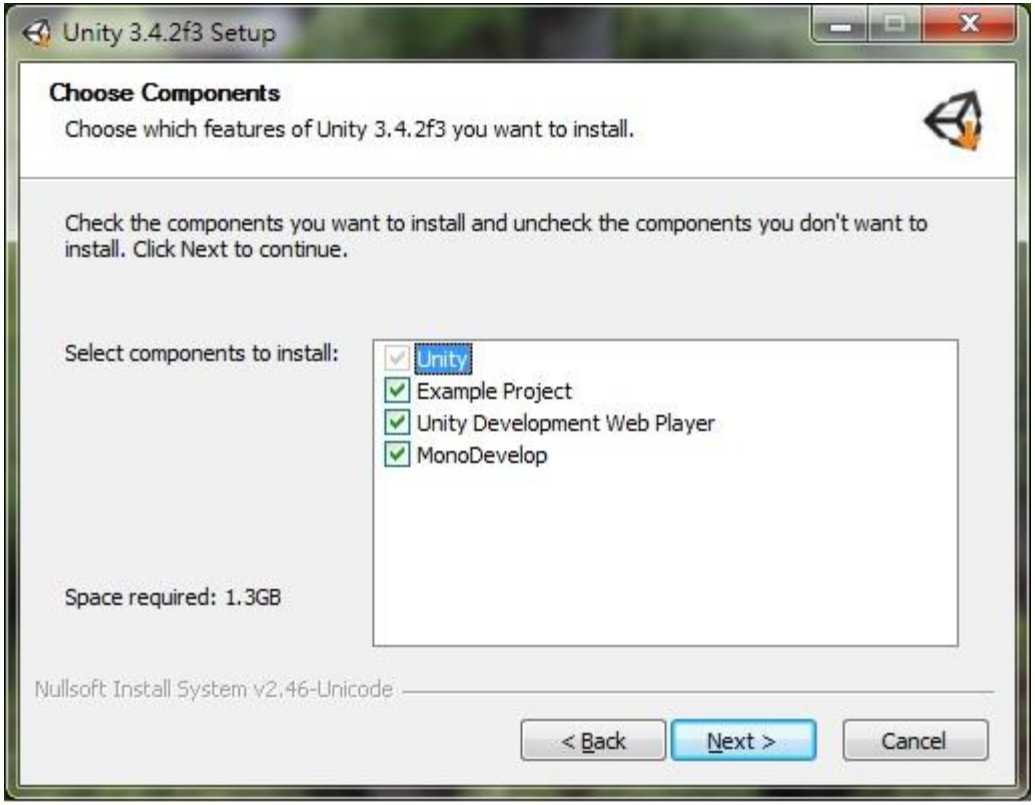
安装Unity


Unity 3.4.2f3 Setup
unity
Welcome to the Unity 3.4.2f3 Setup
Setup will guide you through the installation of Unity 3.4.2f3.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel

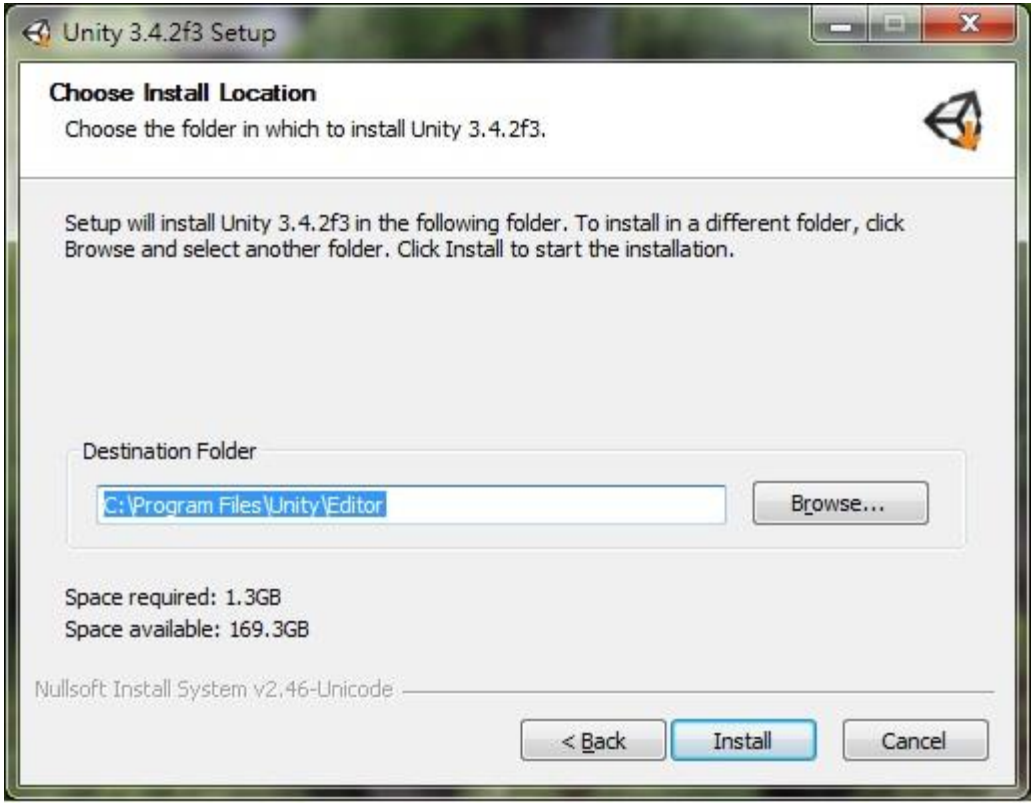
安装Unity


Unity 3.4.2f3 Setup
License Agreement
Please review the license terms before installing Unity 3.4.2f3.
Press Page Down to see the rest of the agreement.
END USER LICENSE AGREEMENT
Software License Agreement for Unity 3.x
PLEASE READ CAREFULLY: BY INSTALLING THE SOFTWARE (AS DEFINED BELOW) AND/OR CLICKING ON THE 'ACCEPT' BUTTON BELOW, YOU (EITHER ON BEHALF OF YOURSELF AS AN INDIVIDUAL OR ON BEHALF OF AN ENTITY AS ITS AUTHORIZED REPRESENTATIVE) AGREE TO ALL OF THE TERMS OF THIS END USER LICENSE AGREEMENT REGARDING THE USE OF THE SOFTWARE. IF YOU DO NOT AGREE WITH ALL OF THE TERMS OF THIS AGREEMENT, CLICK ON THE 'DISAGREE' BUTTON. THIS WILL CANCEL THE INSTALLATION.
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Unity 3.4.2f3.
Nullsoft Install System v2.46-Unicode
< Back
I Agree
Cancel

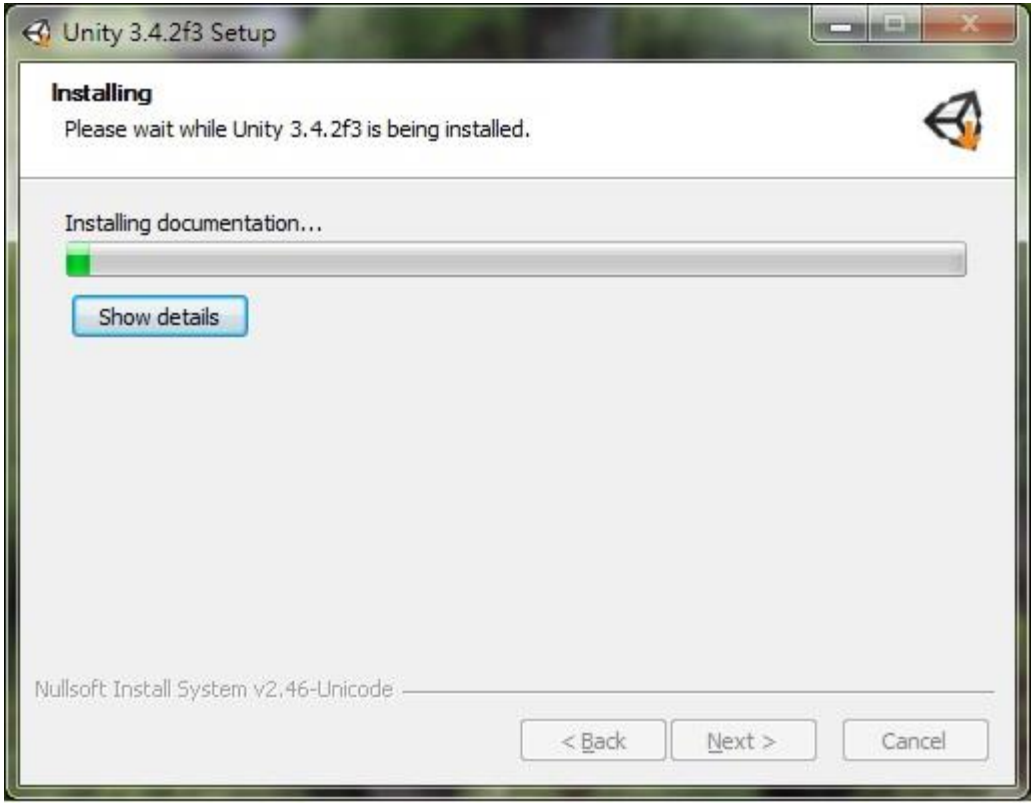
安装Unity


Unity 3.4.2f3 Setup
Choose Components
Choose which features of Unity 3.4.2f3 you want to install.
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.
Select components to install:
Unity
Example Project
Unity Development Web Player
MonoDevelop
Space required: 1.3GB
Nullsoft Install System v2.46-Unicode
< Back
Next >
Cancel

安装Unity


Unity 3.4.2f3 Setup
Choose Install Location
Choose the folder in which to install Unity 3.4.2f3.
Setup will install Unity 3.4.2f3 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.
Destination Folder
C:\Program Files\Unity\Editor
Browse...
Space required: 1.3GB
Space available: 169.3GB
Nullsoft Install System v2.46-Unicode
< Back
Install
Cancel

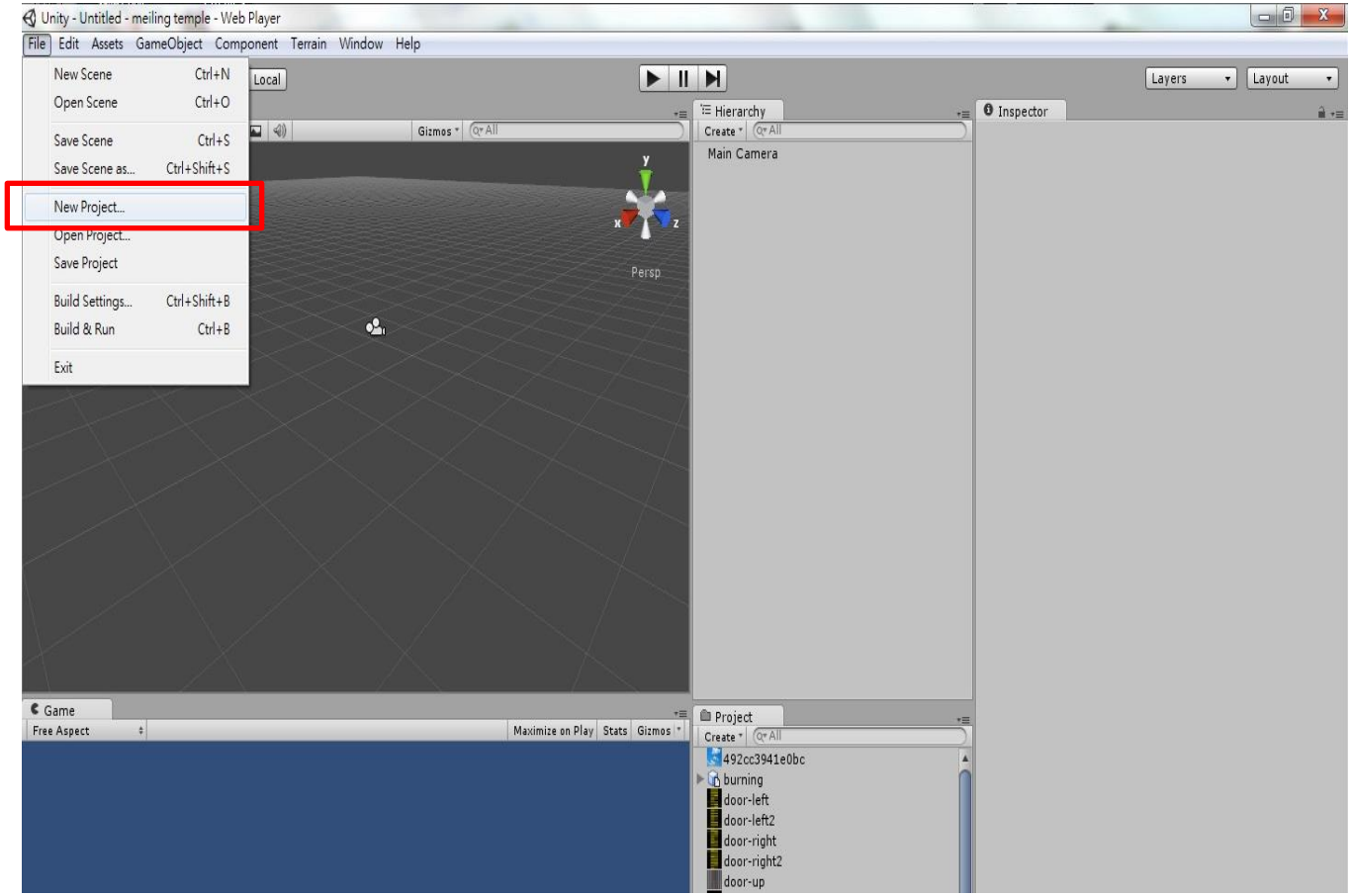
安装Unity


Unity 3.4.2f3 Setup
Installing
Please wait while Unity 3.4.2f3 is being installed.
Installing documentation...
Show details
Nullsoft Install System v2.46-Unicode
< Back
Next >
Cancel

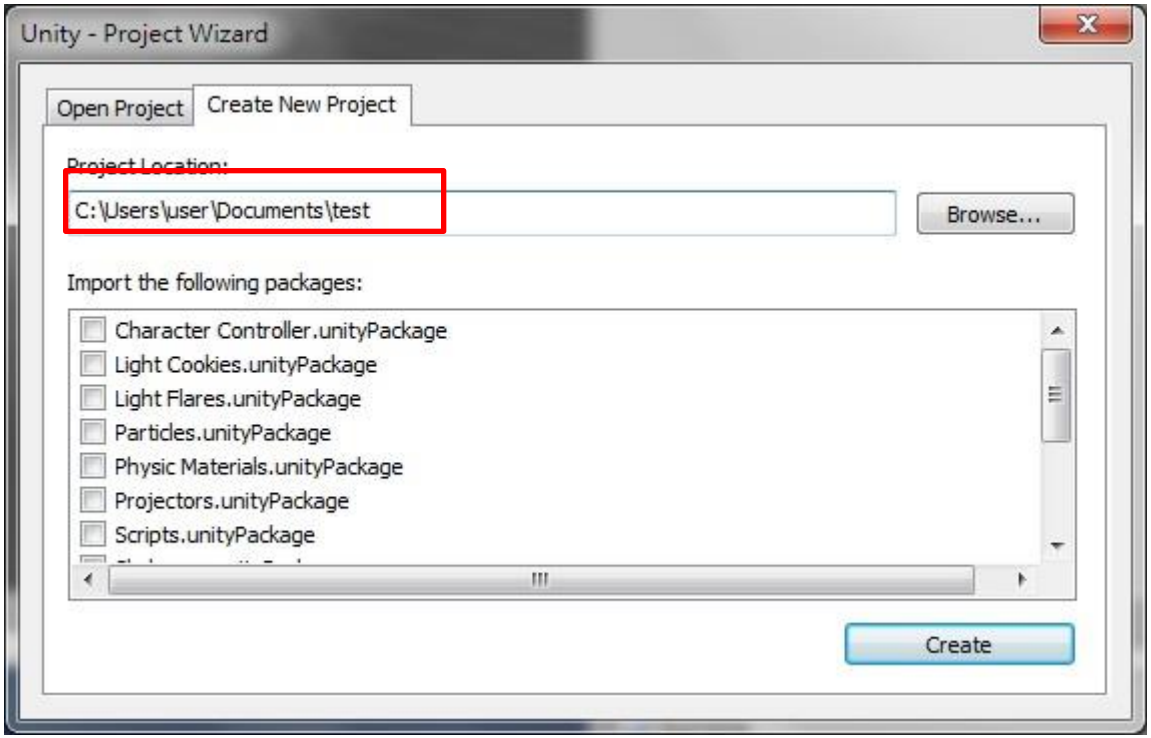
安裝Unity

安裝完成後，需註冊。

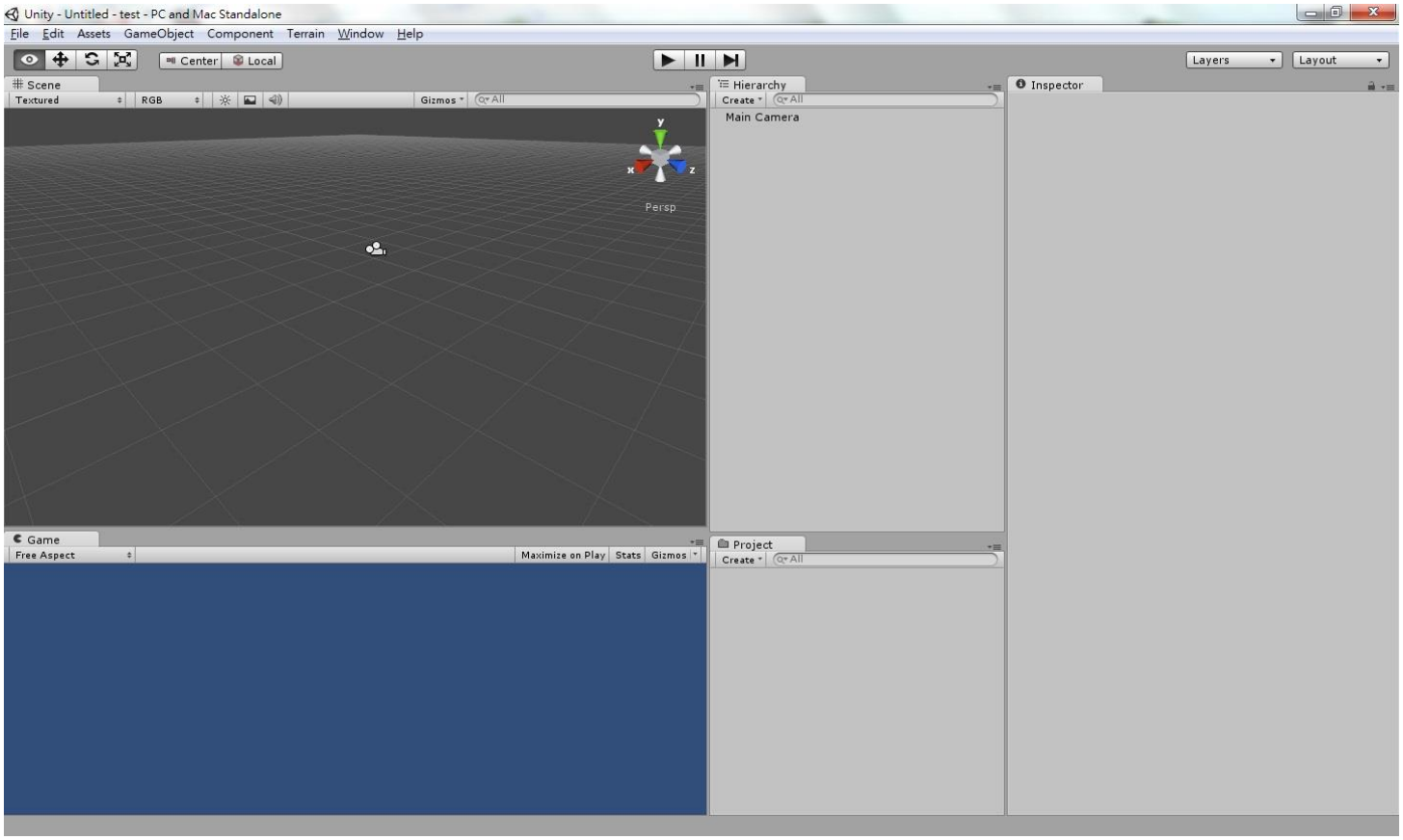
建專案


Unity - Untitled - meiling temple - Web Player
File Edit Assets GameObject Component Terrain Window Help
New Scene Ctrl+N
Open Scene Ctrl+O
Save Scene Ctrl+S
Save Scene as... Ctrl+Shift+S
New Project...
Open Project...
Save Project
Build Settings... Ctrl+Shift+B
Build & Run Ctrl+B
Exit
Local
Layers
Layout
Hierarchy
Create
Main Camera
Inspector
Gizmos
Persp
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Project
Create
492cc3941e0bc
burning
door-left
door-left2
door-right
door-right2
door-up

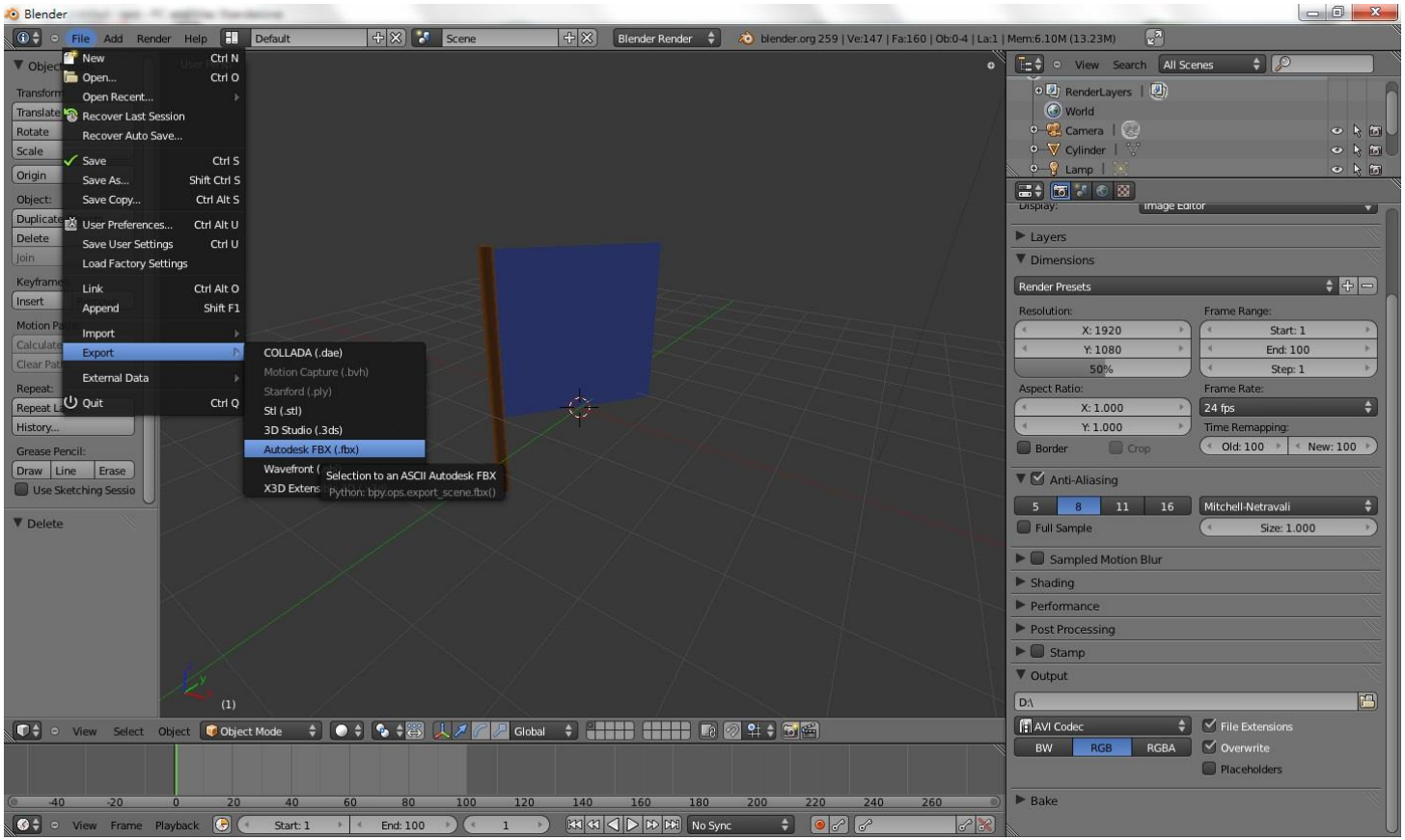
建專案


Unity - Project Wizard
Open Project
Create New Project
Project Location:
C:\Users\user\Documents\test
Browse...
Import the following packages:
Character Controller.unityPackage
Light Cookies.unityPackage
Light Flares.unityPackage
Particles.unityPackage
Physic Materials.unityPackage
Projectors.unityPackage
Scripts.unityPackage
Create

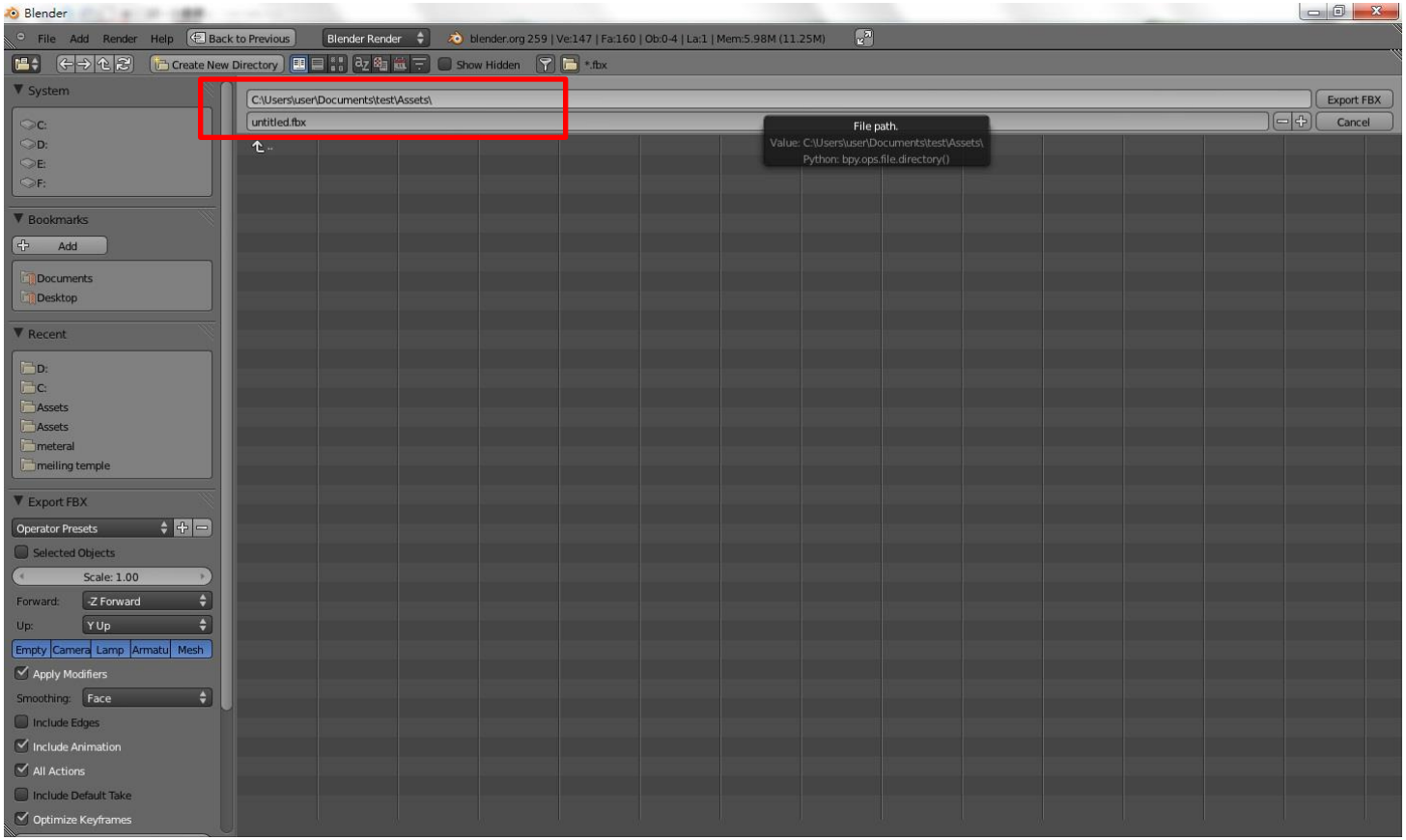
建專案


Unity - Untitled - test - PC and Mac Standalone
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
Inspector
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create All

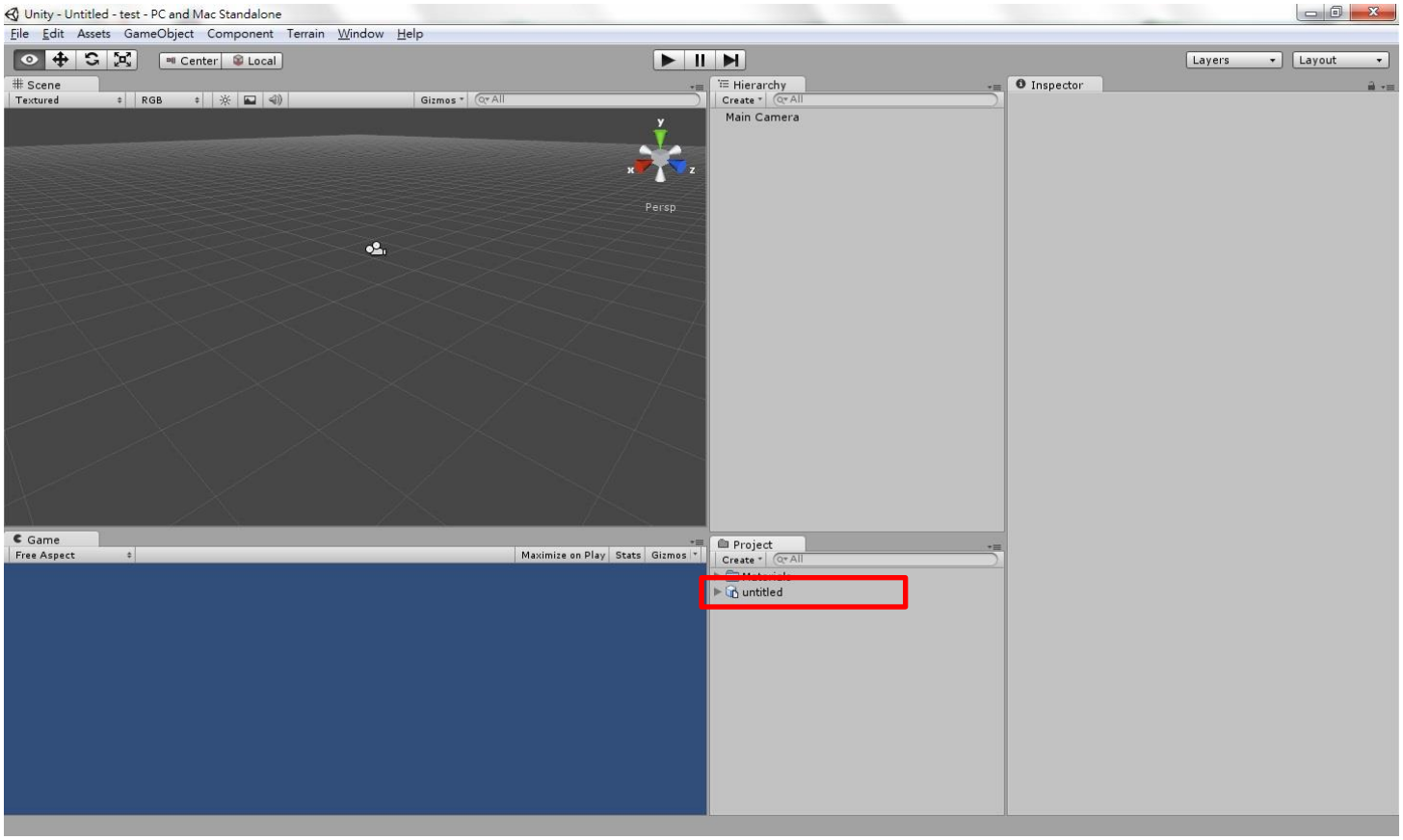
匯入模型


Blender
File Add Render Help
Default
Scene
Blender Render
blender.org 259 | Ve:147 | Fa:160 | Ob:0-4 | La:1 | Mem:6.10M (13.23M)
New Ctrl N
Open... Ctrl O
Open Recent...
Recover Last Session
Recover Auto Save...
Save Ctrl S
Save As... Shift Ctrl S
Save Copy... Ctrl Alt S
User Preferences... Ctrl Alt U
Save User Settings Ctrl U
Load Factory Settings
Link Ctrl Alt O
Append Shift F1
Import
Export
External Data
Quit Ctrl Q
COLLADA (.dae)
Motion Capture (.bvh)
Stanford (.ply)
Stl (.stl)
3D Studio (.3ds)
Autodesk FBX (.fbx)
Wavefront (
X3D Extens
Selection to an ASCII Autodesk FBX
Python: bpy.ops.export_scene.fbx()
View Search All Scenes
RenderLayers
World
Camera
Cylinder
Lamp
Layers
Dimensions
Render Presets
Resolution:
X: 1920
Y: 1080
50%
Frame Range:
Start: 1
End: 100
Step: 1
Aspect Ratio:
X: 1.000
Y: 1.000
Frame Rate:
24 fps
Time Remapping:
Old: 100
New: 100
Border
Crop
Anti-Aliasing
5 8 11 16
Mitchell-Netravali
Full Sample
Size: 1.000
Sampled Motion Blur
Shading
Performance
Post Processing
Stamp
Output
D:\
AVI Codec
File Extensions
BW RGB RGBA
Overwrite
Placeholders
Bake
View Select Object Object Mode
Global
View Frame Playback
Start: 1
End: 100
No Sync

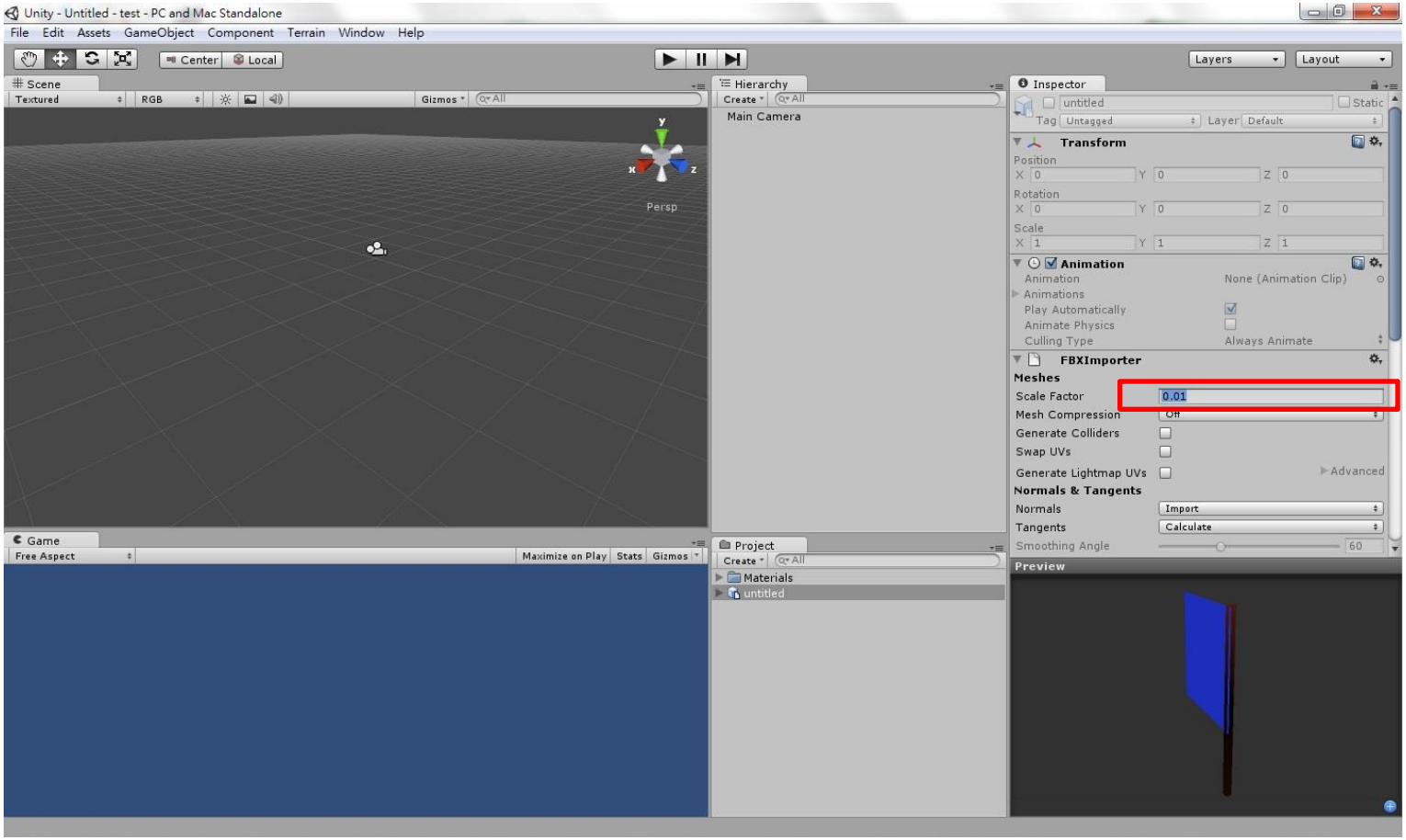
匯入模型


Blender
File Add Render Help Back to Previous Blender Render blender.org 259 | Ve:147 | Fa:160 | Ob:0-4 | La:1 | Mem:5.98M (11.25M)
Create New Directory Show Hidden *.fbx
System
C:
D:
E:
F:
Bookmarks
Add
Documents
Desktop
Recent
D:
C:
Assets
Assets
meteral
meiling temple
Export FBX
Operator Presets
Selected Objects
Scale: 1.00
Forward: -Z Forward
Up: Y Up
Empty Camera Lamp Armatu Mesh
Apply Modifiers
Smoothing: Face
Include Edges
Include Animation
All Actions
Include Default Take
Optimize Keyframes
C:\Users\user\Documents\test\Assets\
untitled.fbx
Export FBX
Cancel
File path.
Value: C:\Users\user\Documents\test\Assets\
Python: bpy.ops.file.directory()

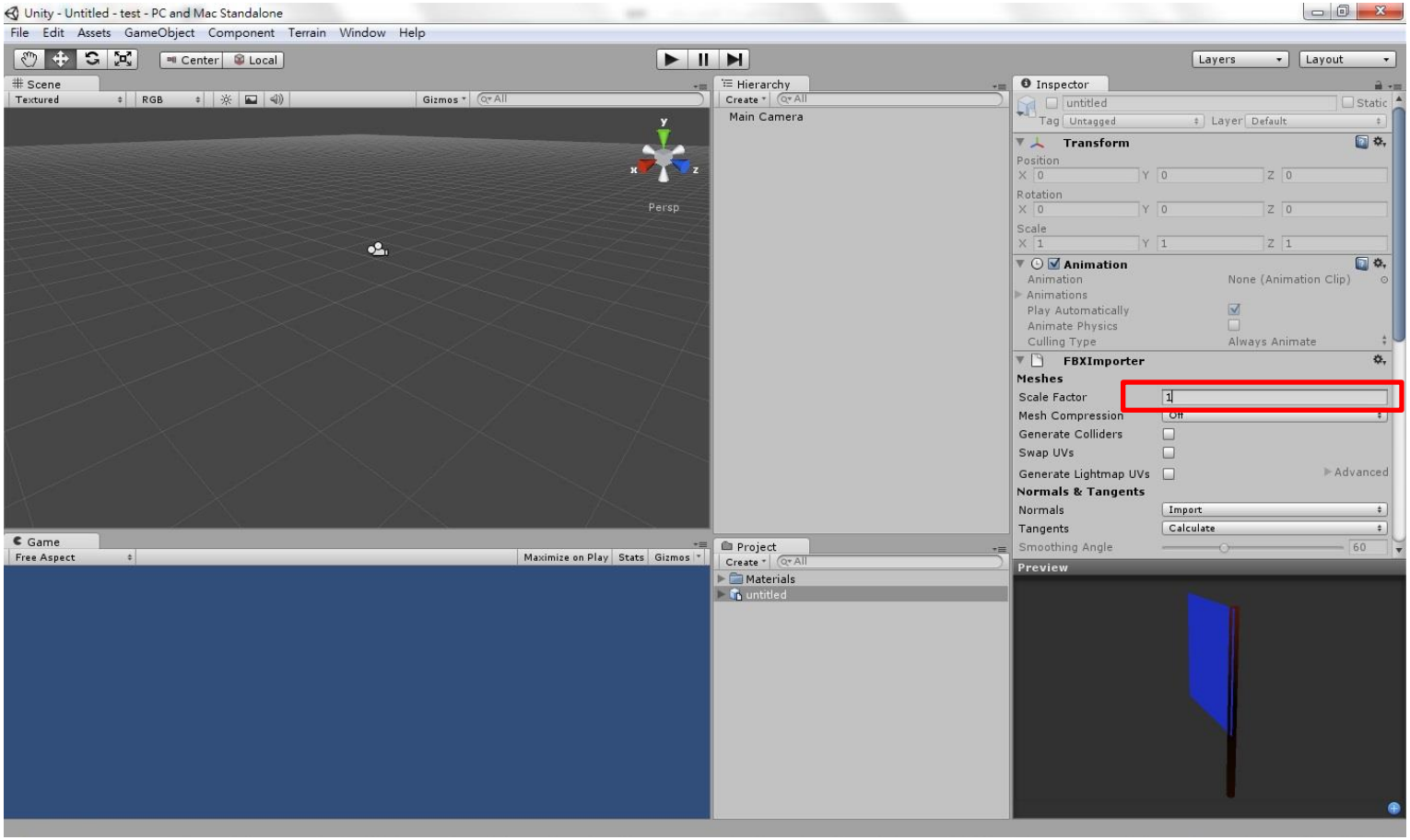
匯入模型


Unity - Untitled - test - PC and Mac Standalone
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
Inspector
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create All
untitled

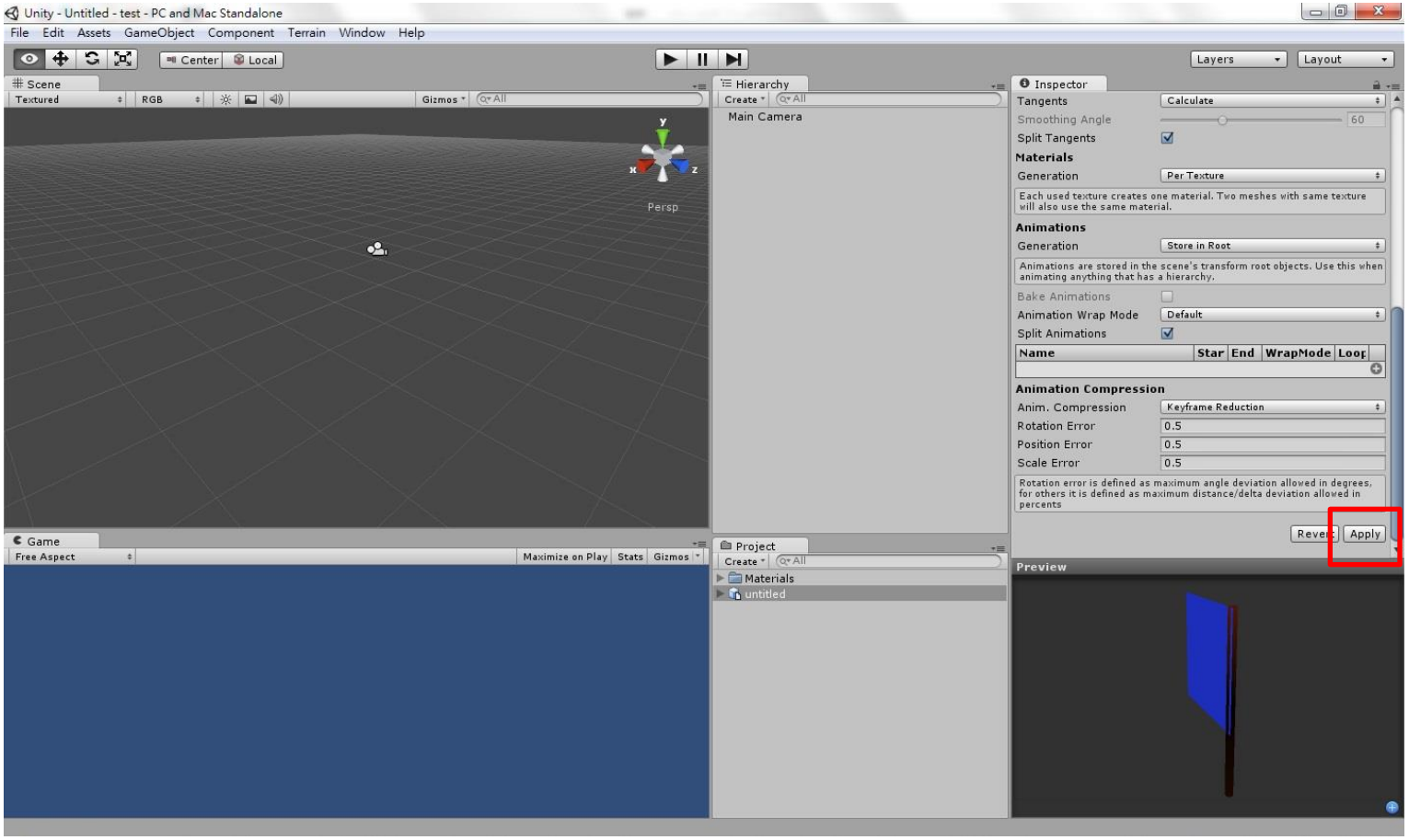
調整大小

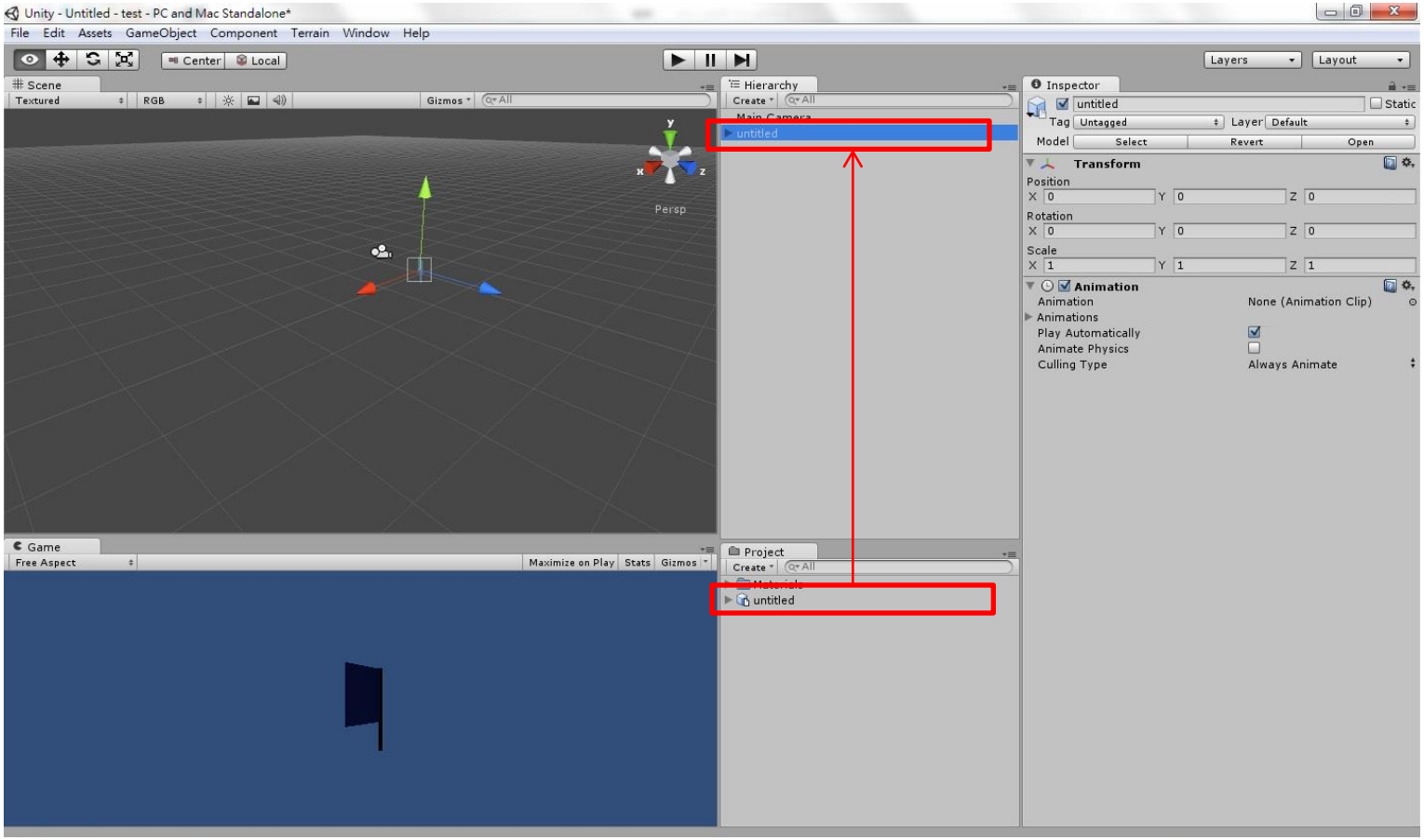

Unity - Untitled - test - PC and Mac Standalone
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
Inspector
untitled
Static
Tag Untagged Layer Default
Transform
Position
X 0 Y 0 Z 0
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Animation
Animation None (Animation Clip)
Animations
Play Automatically
Animate Physics
Culling Type Always Animate
FBXImporter
Meshes
Scale Factor 0.01
Mesh Compression Off
Generate Colliders
Swap UVs
Generate Lightmap UVs Advanced
Normals & Tangents
Normals Import
Tangents Calculate
Smoothing Angle 60
Preview
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create All
Materials
untitled

調整大小

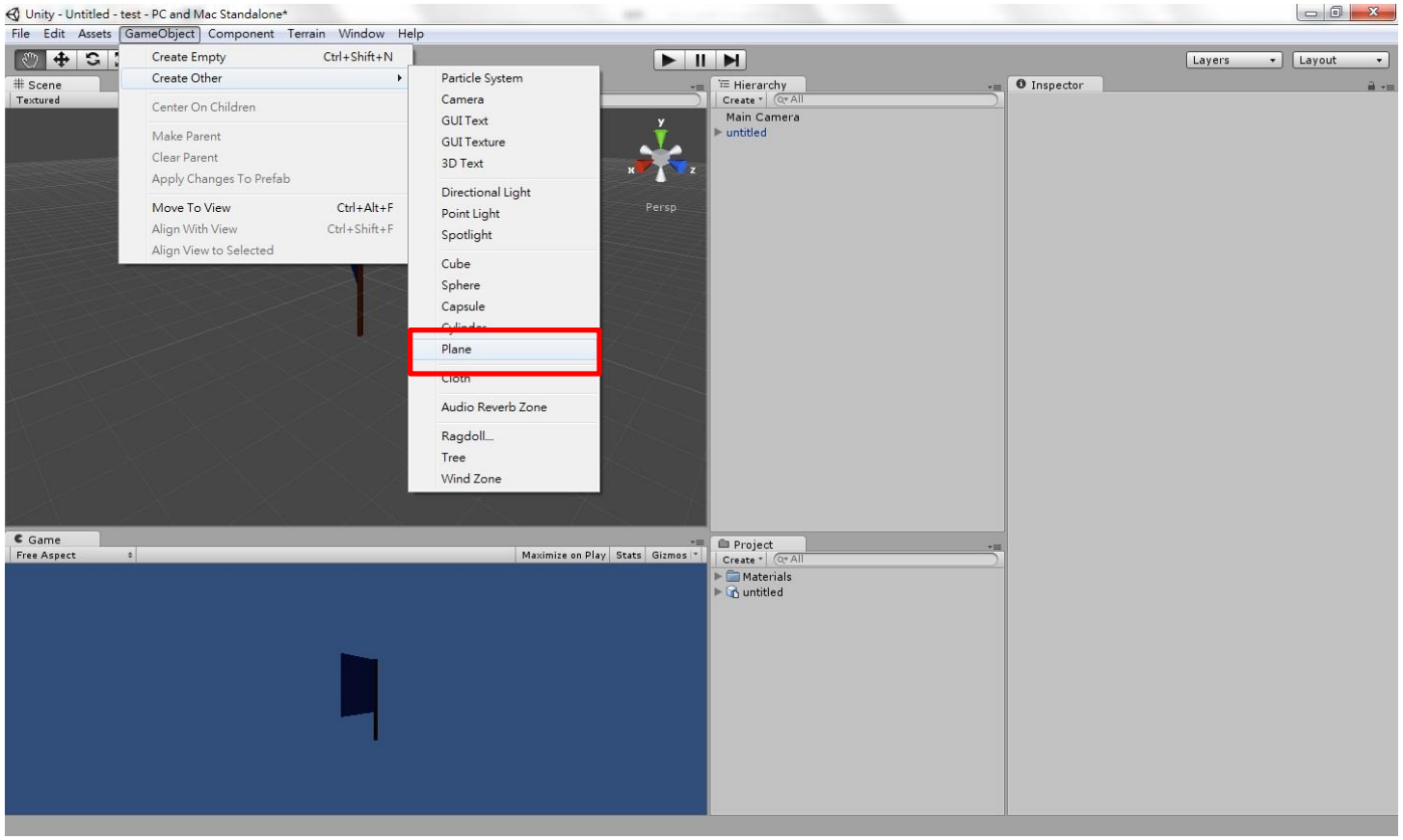

Unity - Untitled - test - PC and Mac Standalone
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
Inspector
untitled
Static
Tag Untagged Layer Default
Transform
Position
X 0 Y 0 Z 0
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Animation
Animation None (Animation Clip)
Animations
Play Automatically
Animate Physics
Culling Type Always Animate
FBXImporter
Meshes
Scale Factor 1
Mesh Compression Off
Generate Colliders
Swap UVs
Generate Lightmap UVs
Advanced
Normals & Tangents
Normals Import
Tangents Calculate
Smoothing Angle 60
Preview
Game
Free Aspect
Maximize on Play Stats Gizmos
Project
Create All
Materials
untitled

調整大小

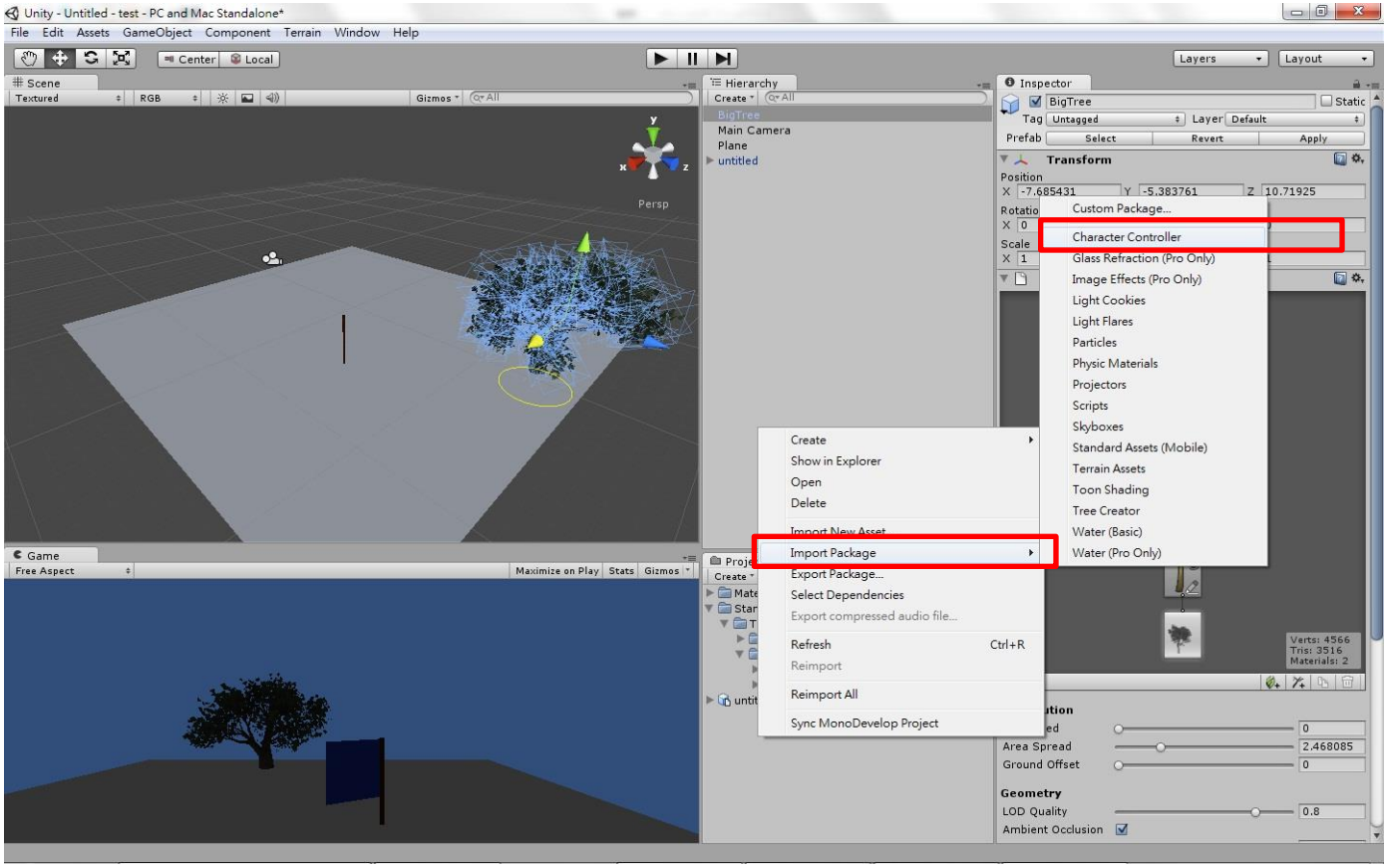

Unity - Untitled - test - PC and Mac Standalone
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
Inspector
Tangents Calculate
Smoothing Angle 60
Split Tangents
Materials
Generation Per Texture
Each used texture creates one material. Two meshes with same texture will also use the same material.
Animations
Generation Store in Root
Animations are stored in the scene's transform root objects. Use this when animating anything that has a hierarchy.
Bake Animations
Animation Wrap Mode Default
Split Animations
Name Star End WrapMode Loop
Animation Compression
Anim. Compression Keyframe Reduction
Rotation Error 0.5
Position Error 0.5
Scale Error 0.5
Rotation error is defined as maximum angle deviation allowed in degrees, for others it is defined as maximum distance/delta deviation allowed in percents
Revert Apply
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create All
Materials
untitled
Preview

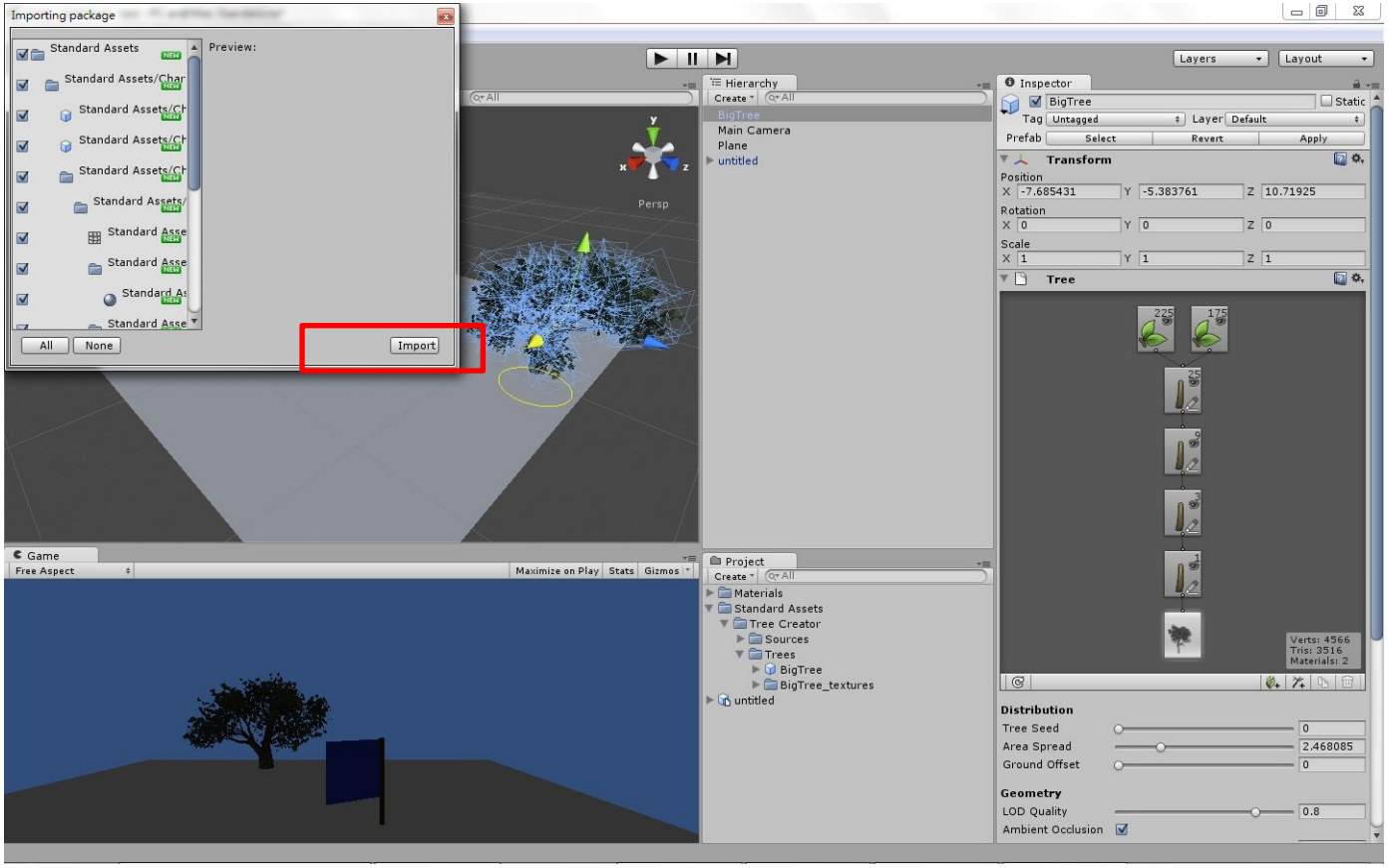

Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos All
Persp
Hierarchy
Create All
Main Camera
untitled
Inspector
untitled
Static
Tag Untagged Layer Default
Model Select Revert Open
Transform
Position
X 0 Y 0 Z 0
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Animation
Animation None (Animation Clip)
Animations
Play Automatically
Animate Physics
Culling Type Always Animate
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create All
untitled

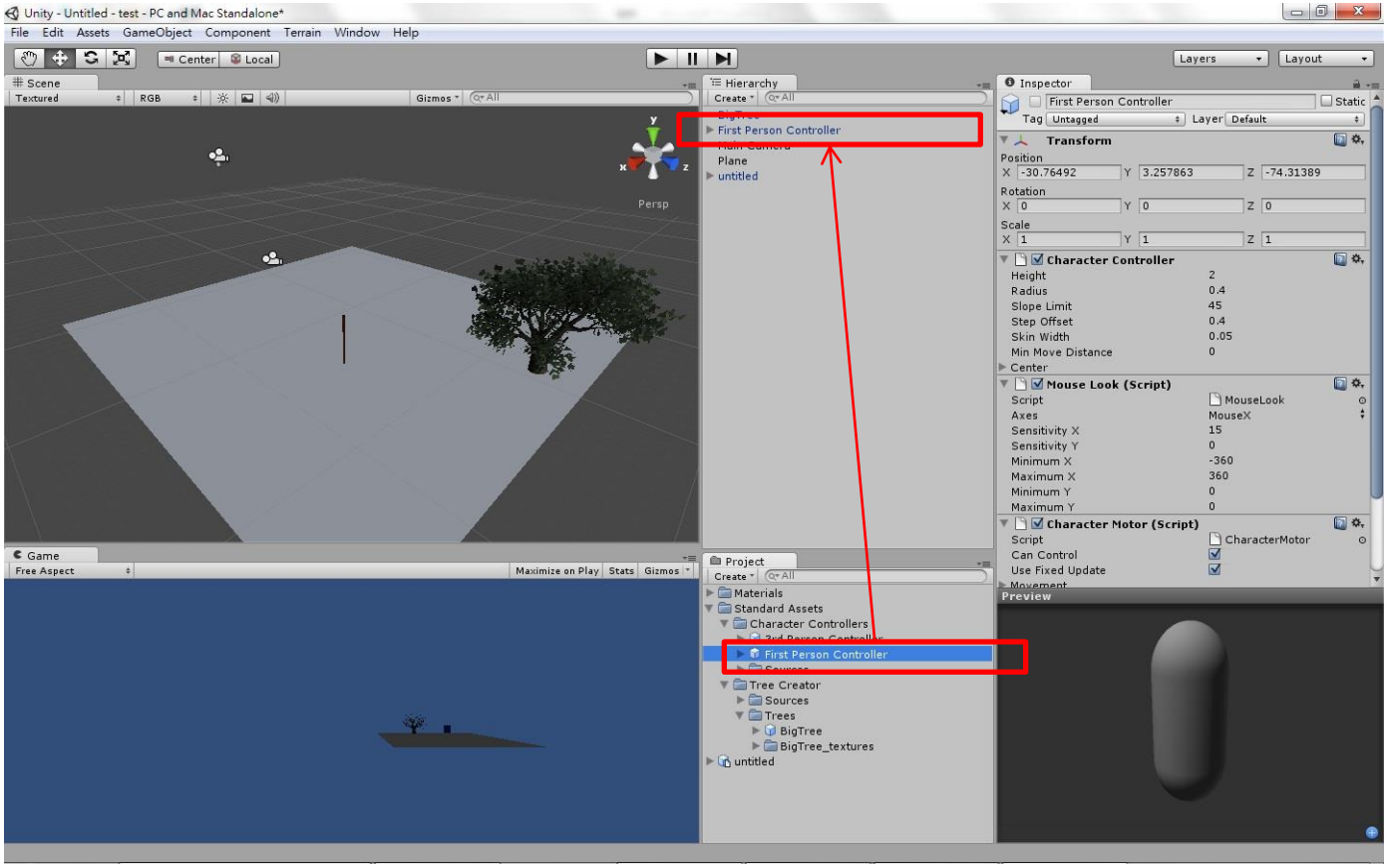
加入Plane

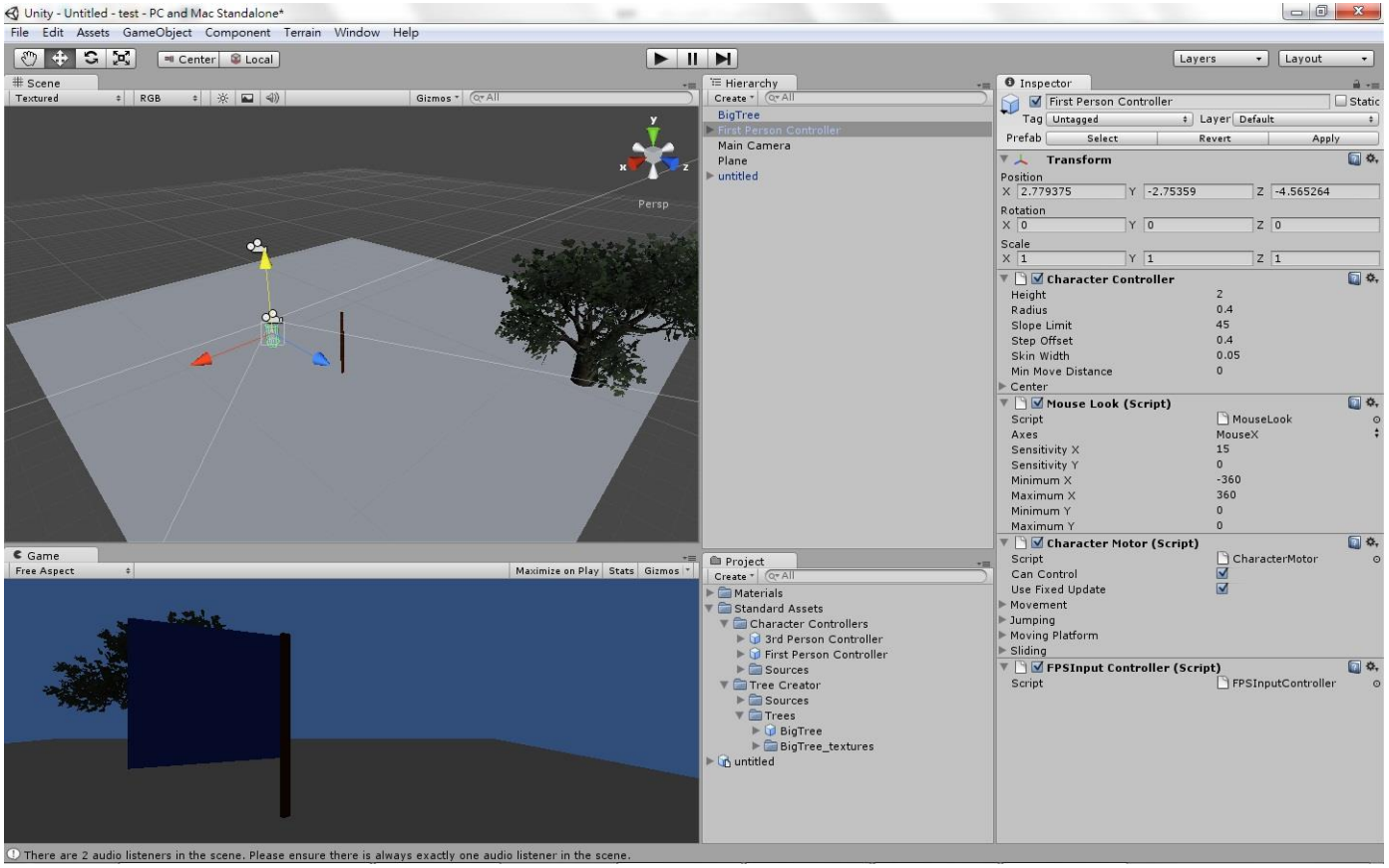

Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Create Empty Ctrl+Shift+N
Create Other
Center On Children
Make Parent
Clear Parent
Apply Changes To Prefab
Move To View Ctrl+Alt+F
Align With View Ctrl+Shift+F
Align View to Selected
Particle System
Camera
GUI Text
GUI Texture
3D Text
Directional Light
Point Light
Spotlight
Cube
Sphere
Capsule
Plane
Cloth
Audio Reverb Zone
Ragdoll...
Tree
Wind Zone
Scene
Textured
Persp
Hierarchy
Create
Main Camera
untitled
Inspector
Layers
Layout
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Project
Create
Materials
untitled

右鍵->Import Package->Character Controller

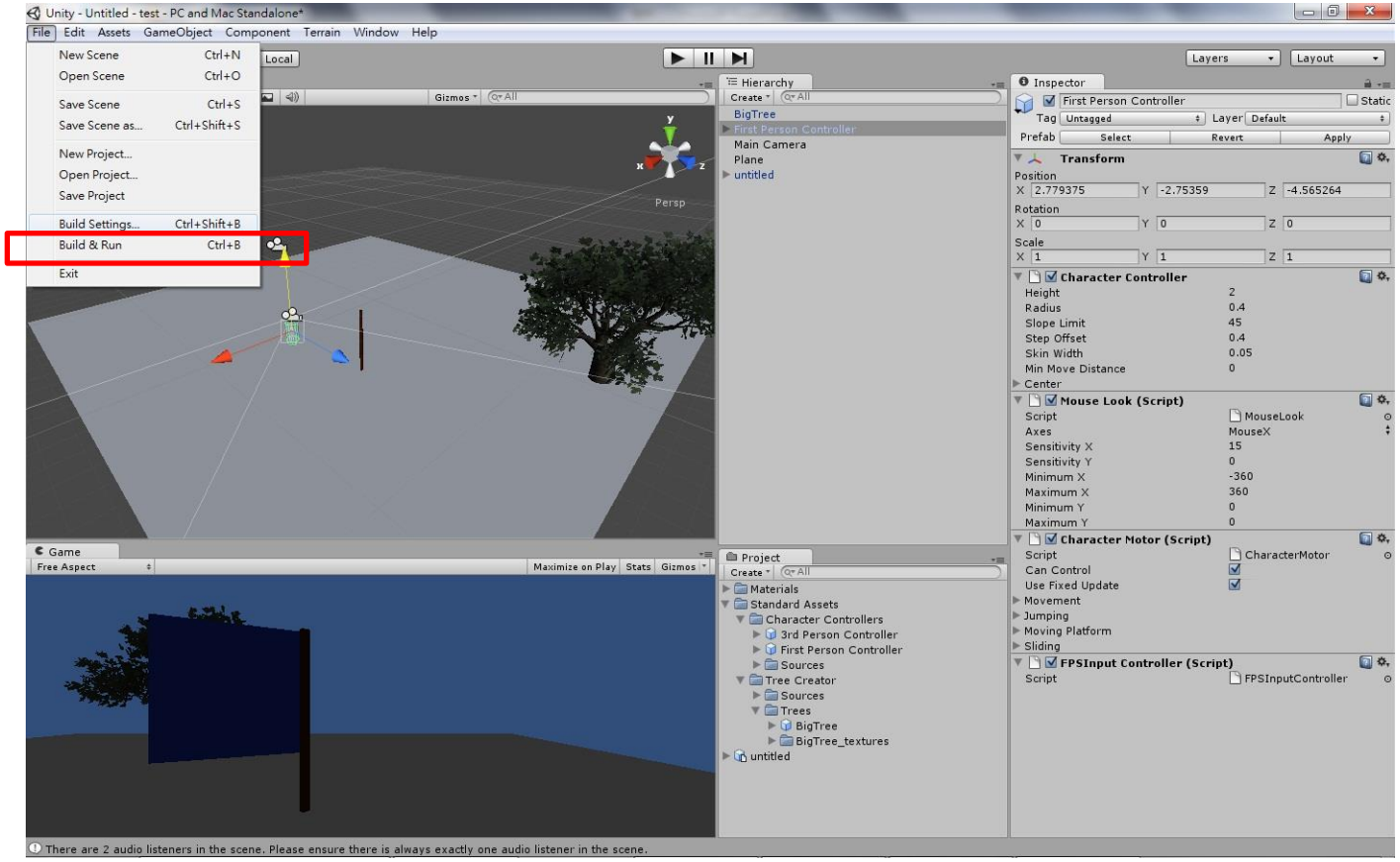


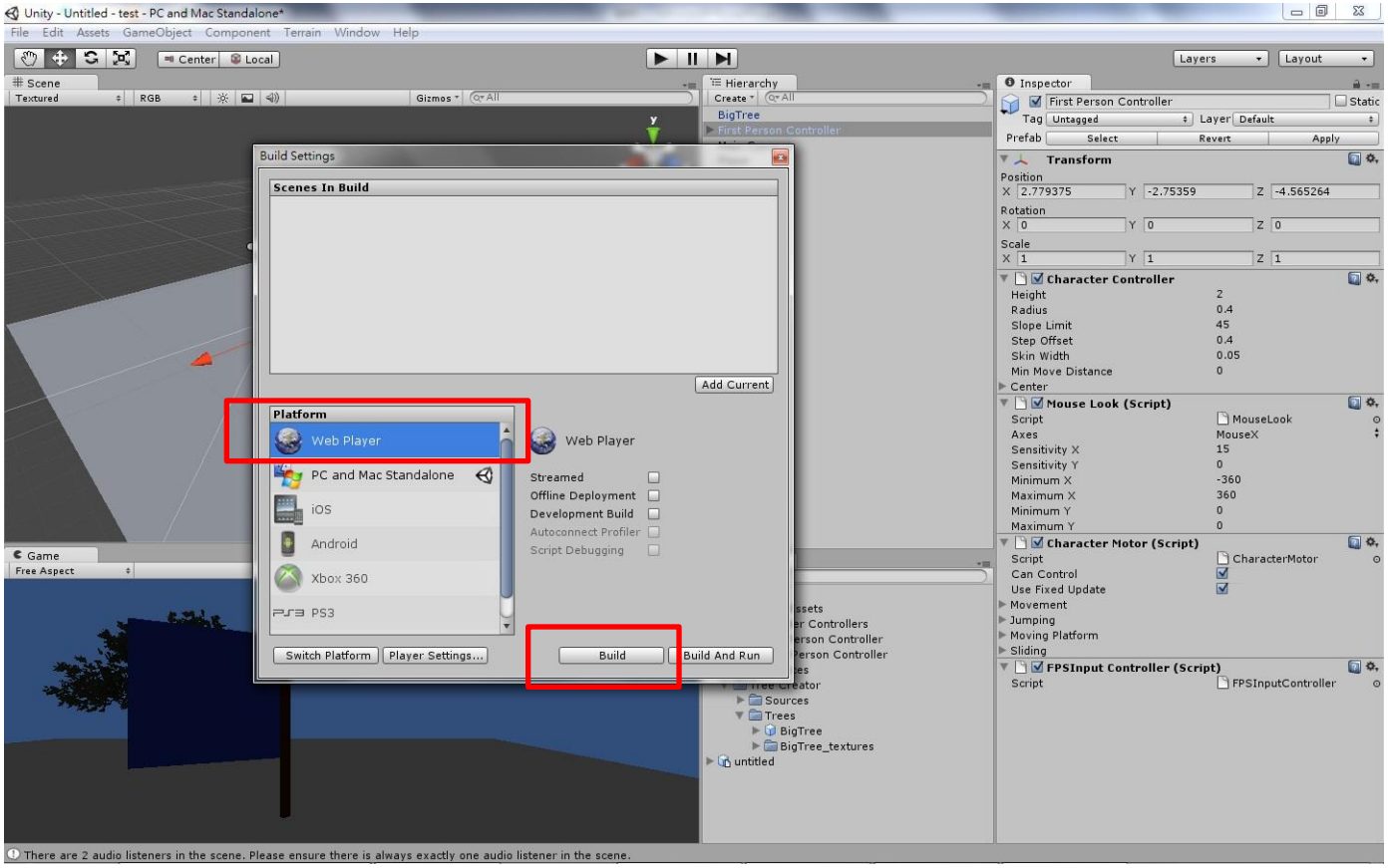

Importing package
Standard Assets
Standard Assets/Char
Standard Assets/Ch
Standard Assets/Ch
Standard Assets/Ch
Standard Assets/
Standard Asse
Standard Asse
Standard As
Standard Asse
Preview:
All
None
Import
Persp
Hierarchy
Create
BigTree
Main Camera
Plane
untitled
Inspector
BigTree
Static
Tag
Untagged
Layer
Default
Prefab
Select
Revert
Apply
Transform
Position
X -7.685431
Y -5.383761
Z 10.71925
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Tree
225
175
25
9
3
1
Verts: 4566
Tris: 3516
Materials: 2
Distribution
Tree Seed 0
Area Spread 2.468085
Ground Offset 0
Geometry
LOD Quality 0.8
Ambient Occlusion
Layers
Layout
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Project
Create
Materials
Standard Assets
Tree Creator
Sources
Trees
BigTree
BigTree_textures
untitled

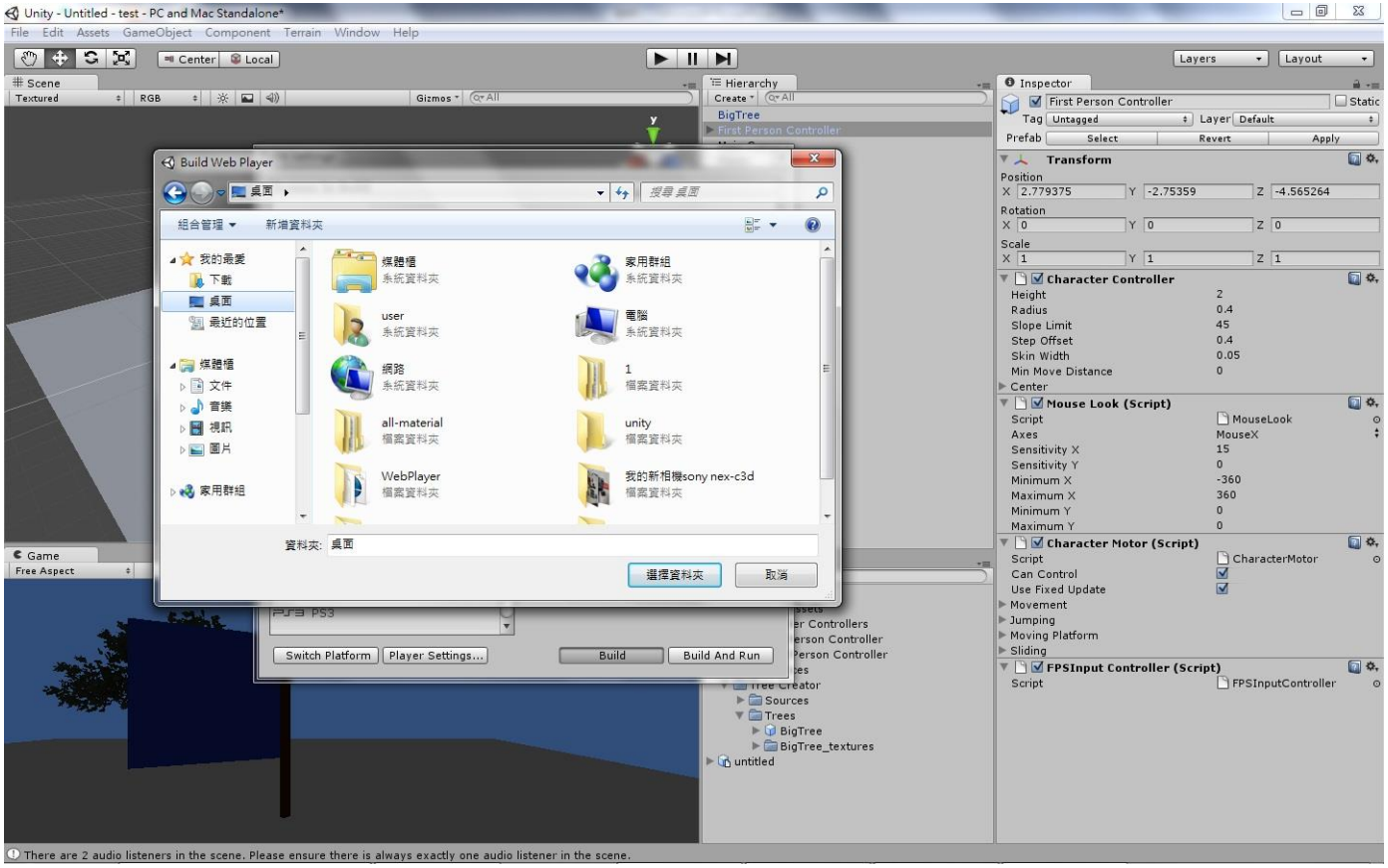

Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos
Persp
Hierarchy
Create
First Person Controller
Plane
untitled
Inspector
First Person Controller
Static
Tag Untagged Layer Default
Transform
Position
X -30.76492 Y 3.257863 Z -74.31389
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Character Controller
Height 2
Radius 0.4
Slope Limit 45
Step Offset 0.4
Skin Width 0.05
Min Move Distance 0
Center
Mouse Look (Script)
Script MouseLook
Axes MouseX
Sensitivity X 15
Sensitivity Y 0
Minimum X -360
Maximum X 360
Minimum Y 0
Maximum Y 0
Character Motor (Script)
Script CharacterMotor
Can Control
Use Fixed Update
Movement
Preview
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create
Materials
Standard Assets
Character Controllers
First Person Controller
Tree Creator
Sources
Trees
BigTree
BigTree_textures
untitled

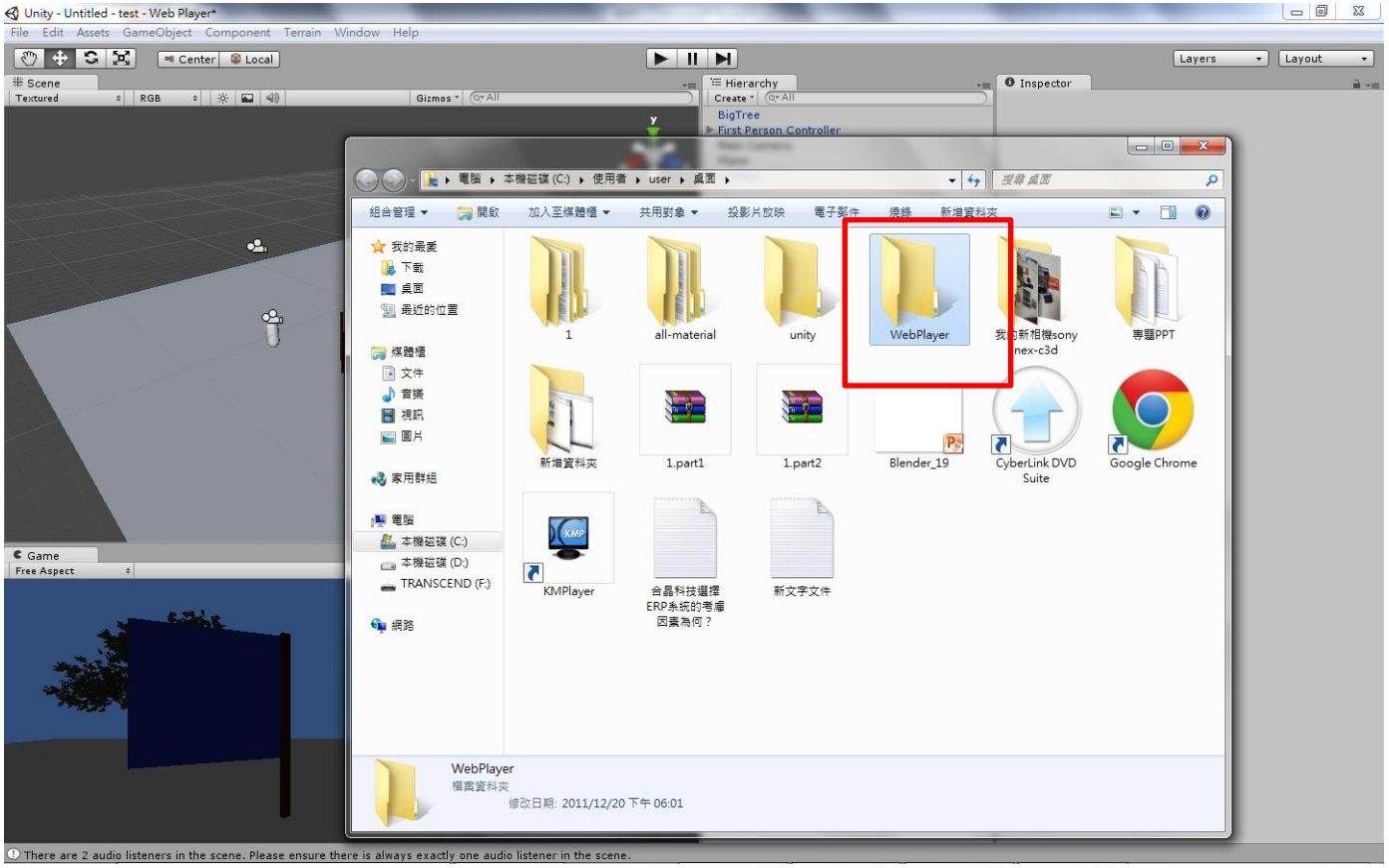

Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos
Persp
Hierarchy
Create
BigTree
First Person Controller
Main Camera
Plane
untitled
Inspector
First Person Controller
Static
Tag Untagged Layer Default
Prefab Select Revert Apply
Transform
Position
X 2.779375 Y -2.75359 Z -4.565264
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Character Controller
Height 2
Radius 0.4
Slope Limit 45
Step Offset 0.4
Skin Width 0.05
Min Move Distance 0
Center
Mouse Look (Script)
Script MouseLook
Axes MouseX
Sensitivity X 15
Sensitivity Y 0
Minimum X -360
Maximum X 360
Minimum Y 0
Maximum Y 0
Character Motor (Script)
Script CharacterMotor
Can Control
Use Fixed Update
Movement
Jumping
Moving Platform
Sliding
FPSInput Controller (Script)
Script FPSInputController
Game
Free Aspect Maximize on Play Stats Gizmos
Project
Create
Materials
Standard Assets
Character Controllers
3rd Person Controller
First Person Controller
Sources
Tree Creator
Sources
Trees
BigTree
BigTree_textures
untitled
There are 2 audio listeners in the scene. Please ensure there is always exactly one audio listener in the scene.

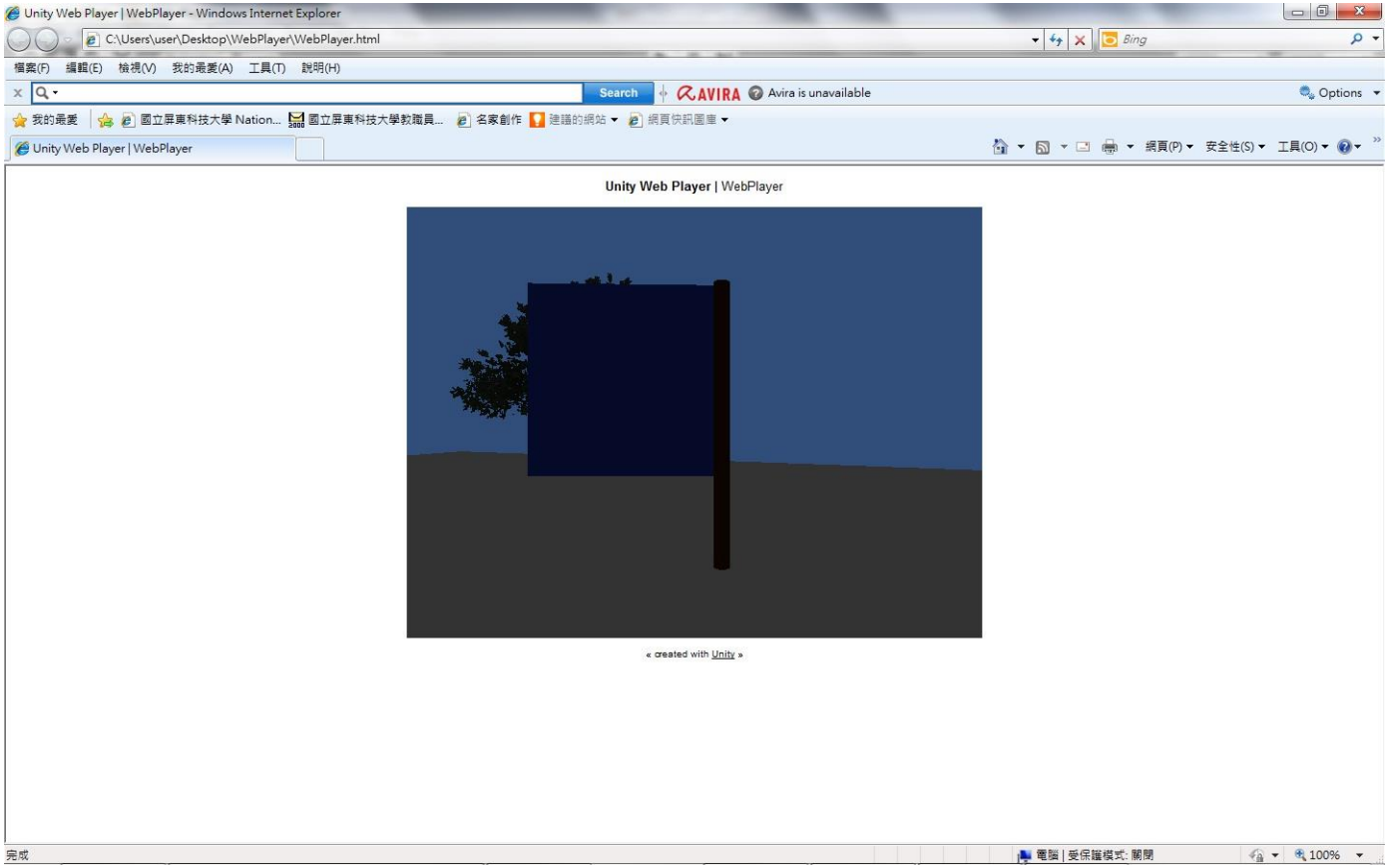
匯出


Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
New Scene Ctrl+N
Open Scene Ctrl+O
Save Scene Ctrl+S
Save Scene as... Ctrl+Shift+S
New Project...
Open Project...
Save Project
Build Settings... Ctrl+Shift+B
Build & Run Ctrl+B
Exit
Hierarchy
BigTree
First Person Controller
Main Camera
Plane
untitled
Inspector
First Person Controller
Transform
Character Controller
Mouse Look (Script)
Character Motor (Script)
FPSInput Controller (Script)
Game
Project
There are 2 audio listeners in the scene. Please ensure there is always exactly one audio listener in the scene.


Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos
Hierarchy
Create
BigTree
First Person Controller
Inspector
First Person Controller
Static
Tag Untagged Layer Default
Prefab Select Revert Apply
Transform
Position
X 2.779375 Y -2.75359 Z -4.565264
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Character Controller
Height 2
Radius 0.4
Slope Limit 45
Step Offset 0.4
Skin Width 0.05
Min Move Distance 0
Center
Mouse Look (Script)
Script MouseLook
Axes MouseX
Sensitivity X 15
Sensitivity Y 0
Minimum X -360
Maximum X 360
Minimum Y 0
Maximum Y 0
Character Motor (Script)
Script CharacterMotor
Can Control
Use Fixed Update
Movement
Jumping
Moving Platform
Sliding
FPSInput Controller (Script)
Script FPSInputController
Build Settings
Scenes In Build
Add Current
Platform
Web Player
PC and Mac Standalone
iOS
Android
Xbox 360
PS3
Web Player
Streamed
Offline Deployment
Development Build
Autoconnect Profiler
Script Debugging
Switch Platform
Player Settings...
Build
Build And Run
Game
Free Aspect
Sources
Trees
BigTree
BigTree_textures
untitled
There are 2 audio listeners in the scene. Please ensure there is always exactly one audio listener in the scene.


Unity - Untitled - test - PC and Mac Standalone*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos
Hierarchy
Create
BigTree
First Person Controller
Inspector
First Person Controller
Static
Tag Untagged Layer Default
Prefab Select Revert Apply
Transform
Position
X 2.779375 Y -2.75359 Z -4.565264
Rotation
X 0 Y 0 Z 0
Scale
X 1 Y 1 Z 1
Character Controller
Height 2
Radius 0.4
Slope Limit 45
Step Offset 0.4
Skin Width 0.05
Min Move Distance 0
Center
Mouse Look (Script)
Script MouseLook
Axes MouseX
Sensitivity X 15
Sensitivity Y 0
Minimum X -360
Maximum X 360
Minimum Y 0
Maximum Y 0
Character Motor (Script)
Script CharacterMotor
Can Control
Use Fixed Update
Movement
Jumping
Moving Platform
Sliding
FPSInput Controller (Script)
Script FPSInputController
Build Web Player
桌面
搜尋 桌面
組合管理
新增資料夾
我的最愛
下載
桌面
最近的位置
媒體櫃
文件
音樂
視訊
圖片
家用群組
媒體櫃
系統資料夾
家用群組
系統資料夾
user
系統資料夾
電腦
系統資料夾
網路
系統資料夾
1
檔案資料夾
all-material
檔案資料夾
unity
檔案資料夾
WebPlayer
檔案資料夾
我的新相機sony nex-c3d
檔案資料夾
資料夾: 桌面
選擇資料夾
取消
Switch Platform
Player Settings...
Build
Build And Run
Game
Free Aspect
Sources
Trees
BigTree
BigTree_textures
untitled
There are 2 audio listeners in the scene. Please ensure there is always exactly one audio listener in the scene.


Unity - Untitled - test - Web Player*
File Edit Assets GameObject Component Terrain Window Help
Center Local
Layers Layout
Scene
Textured RGB Gizmos
Hierarchy
Create
BigTree
First Person Controller
Inspector
Game
Free Aspect
There are 2 audio listeners in the scene. Please ensure there is always exactly one audio listener in the scene.
電腦 本機磁碟 (C:) 使用者 user 桌面
組合管理 開啟 加入至媒體櫃 共用對象 投影片放映 電子郵件 燒錄 新增資料夾
我的最愛 下載 桌面 最近的位置
媒體櫃 文件 音樂 視訊 圖片
家用群組
電腦 本機磁碟 (C:) 本機磁碟 (D:) TRANSCEND (F:)
網路
1
all-material
unity
WebPlayer
專題PPT
新增資料夾
1.part1
1.part2
Blender_19
CyberLink DVD Suite
Google Chrome
KMPlayer
合晶科技選擇ERP系統的考慮因素為何？
新文字文件
WebPlayer
檔案資料夾
修改日期: 2011/12/20 下午 06:01


Unity Web Player | WebPlayer - Windows Internet Explorer
C:\Users\user\Desktop\WebPlayer\WebPlayer.html
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
Search
Avira is unavailable
Options
我的最愛
Unity Web Player | WebPlayer
網頁(P) 安全性(S) 工具(O)
Unity Web Player | WebPlayer
« created with Unity »
完成
電腦 | 受保護模式: 關閉
100%

單元二 動態效果

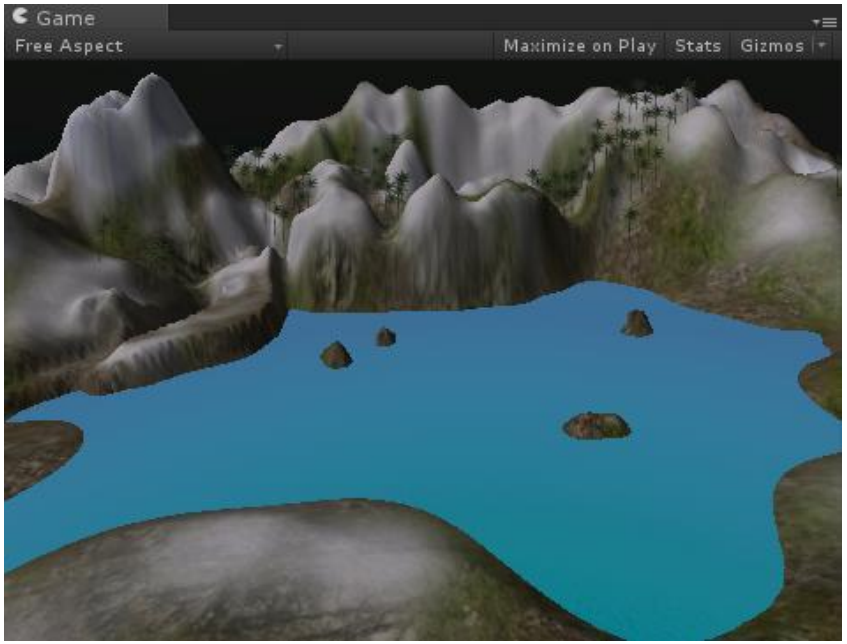
遊戲場景中水特效及粒子動態效果

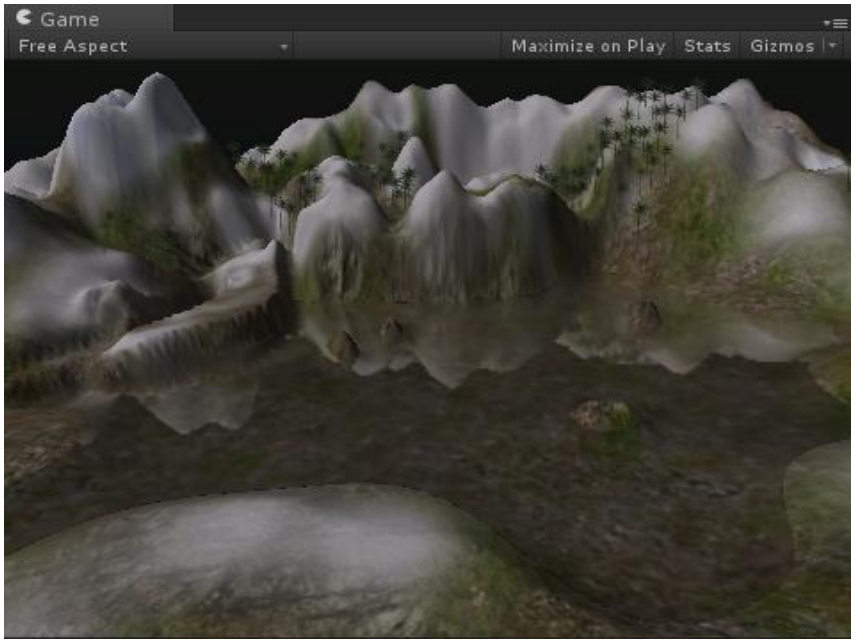
學習重點

- 在場景中利用水資源包建立水特效
- 在場景中建立動態粒子特效

在場景中利用水資源包建立水特效

Water(Basic)　　Water(Pro Only)

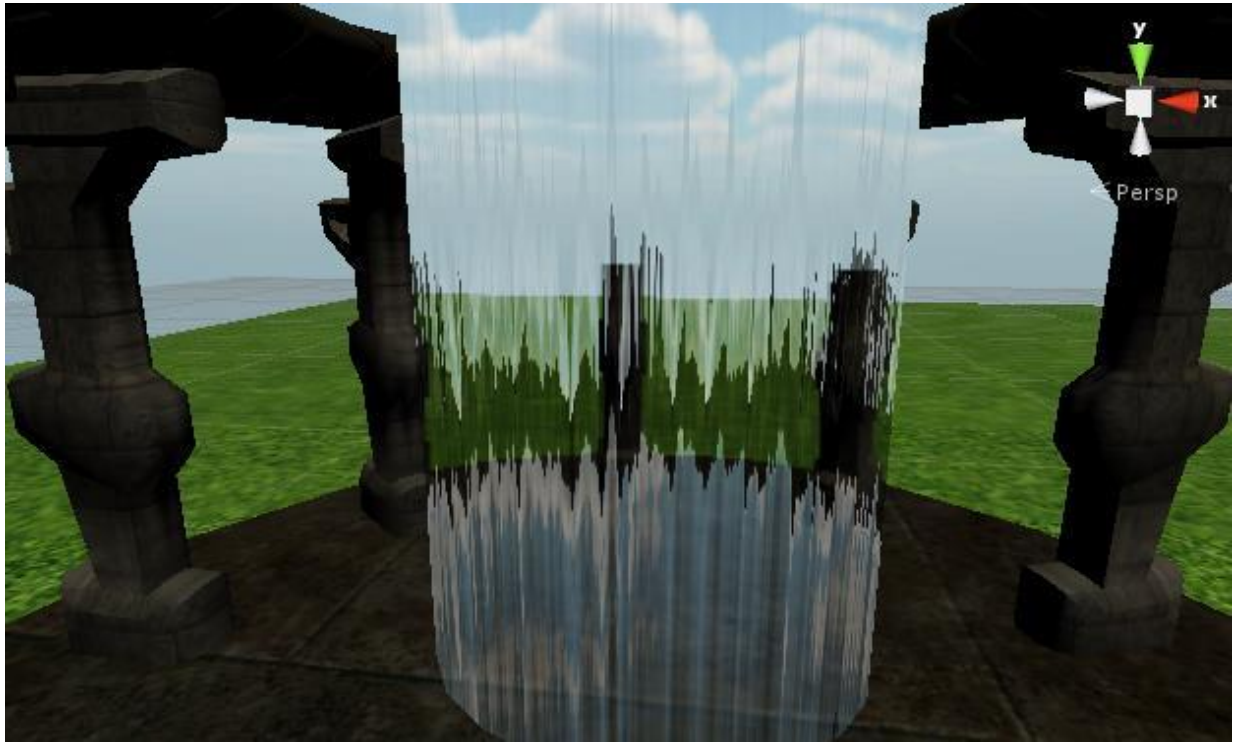




- 能對遊戲場景中的天空與物體進行反射或折射運算並產生水波效果。

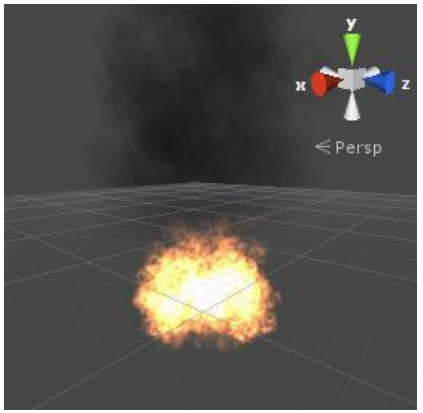
- 可以同時在場景中放置多個水的區域，不同的水區域會因為位置高低與旋轉角度的不同而反射出不同的倒影。

- 水區域的形狀有多樣的變化。



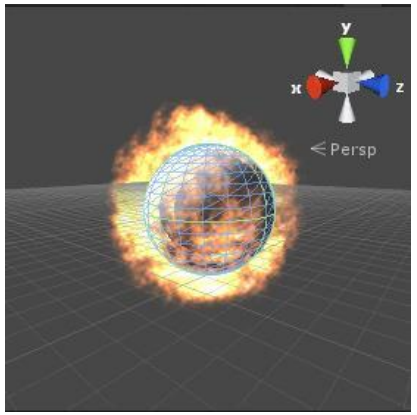
在場景中建立動態粒子特效

- 利用二維的圖片經由不斷重複的生成，進而在三維的空間中產生的動態效果。



- 粒子系統

Ellipsoed Particle(橢球粒子發射器)
Mesh Paticle Emitter(網格粒子發射器)
Particle Animator(粒子動畫器)
Particle Render(粒子渲染器)
World Particle Collider(粒子碰撞器)



單元三 建置地形

創建遊戲基本地形

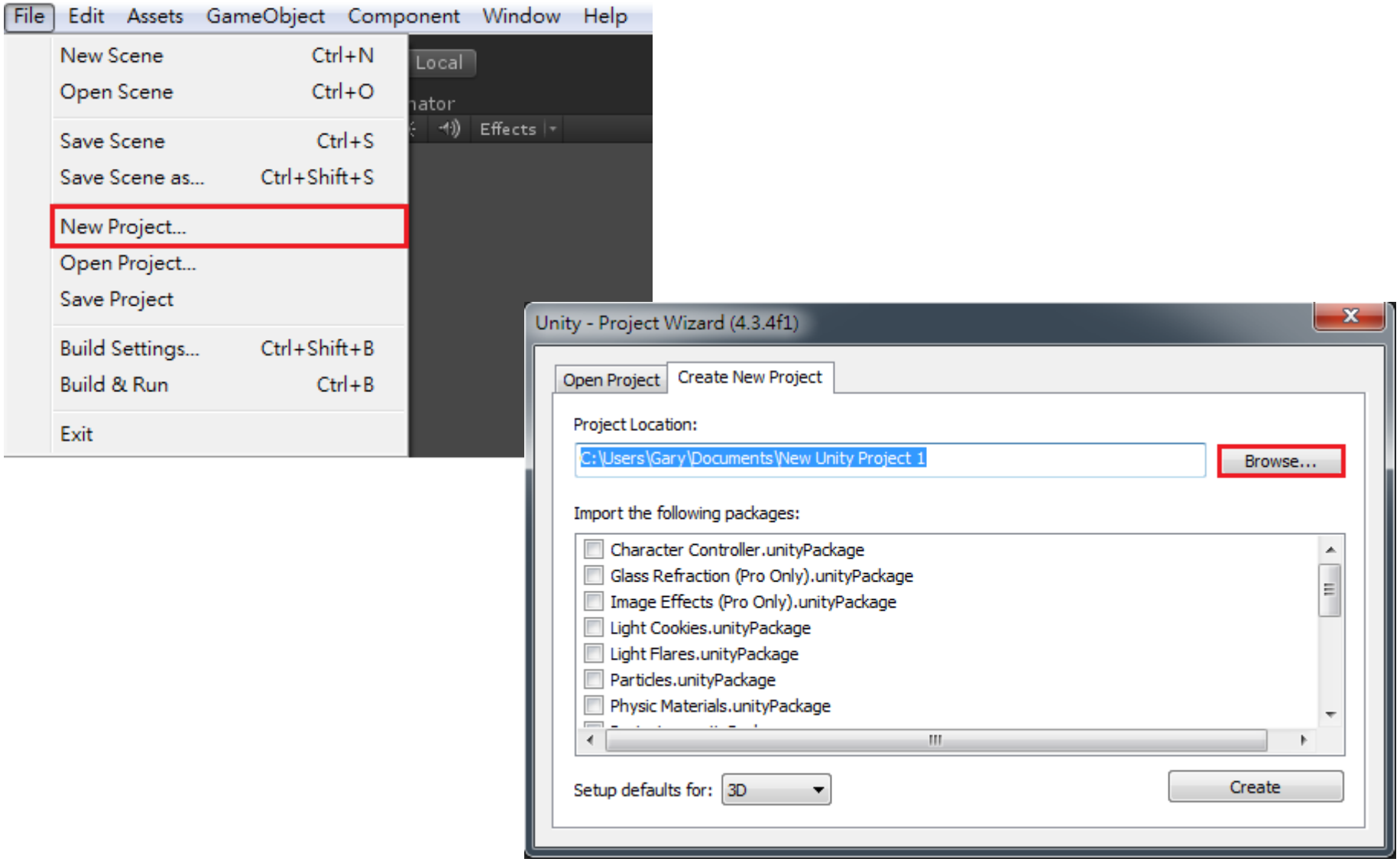
本範例主要的學習重點

- 重點一:建立遊戲專案。
- 重點二:使用地形編輯器建立遊戲場景地形。
- 重點三:建立光源與第一人稱控制器。

重點一:建立遊戲專案

- 任何作品利用Unity來創作的開始都是先建立專案，透過專案的建立方式，在相同專案底下可以彼此共享資源。對已經建立的專案在專案的視窗可以查詢專案名稱以及專案存放的路徑。

如何創造一個新的專案


File
Edit
Assets
GameObject
Component
Window
Help
New Scene Ctrl+N
Open Scene Ctrl+O
Save Scene Ctrl+S
Save Scene as... Ctrl+Shift+S
New Project...
Open Project...
Save Project
Build Settings... Ctrl+Shift+B
Build & Run Ctrl+B
Exit
Local
Effects
Unity - Project Wizard (4.3.4f1)
Open Project
Create New Project
Project Location:
C:\Users\Gary\Documents\New Unity Project 1
Browse...
Import the following packages:
Character Controller.unityPackage
Glass Refraction (Pro Only).unityPackage
Image Effects (Pro Only).unityPackage
Light Cookies.unityPackage
Light Flares.unityPackage
Particles.unityPackage
Physic Materials.unityPackage
Setup defaults for:
3D
Create

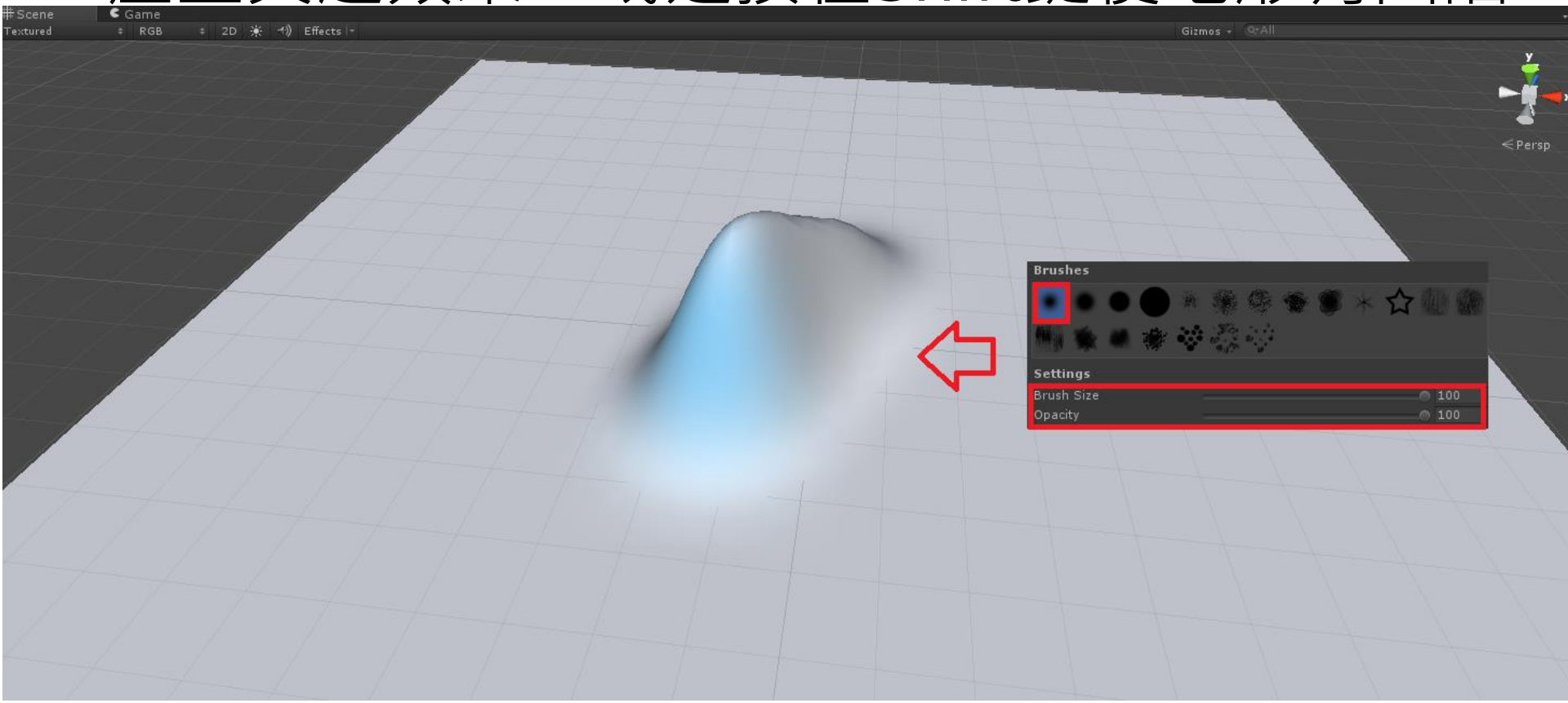
重點二: 使用地形編輯器建立遊戲場景地形

- 遊戲的場景地形可以從外部製作完成再匯入Unity來使用，也可以利用Unity中的Terrain Assets(地形編輯器)來製作遊戲場景所需要的地形。

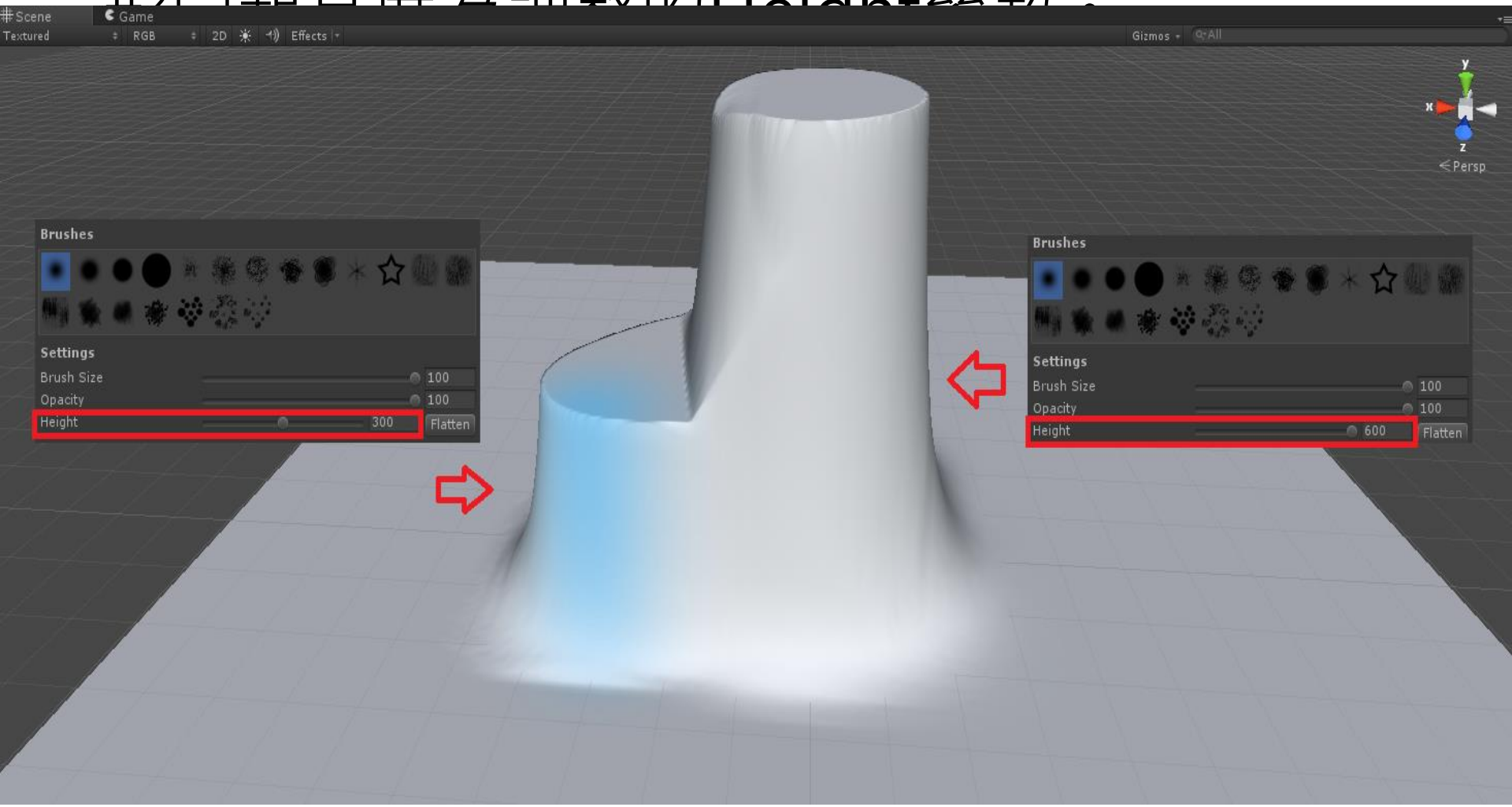
Terrain Assets(地形編輯器)

- 由左至右依序為凹凸地形、繪製高度、平滑地形、材質繪製、種植樹木、繪製細節、設定

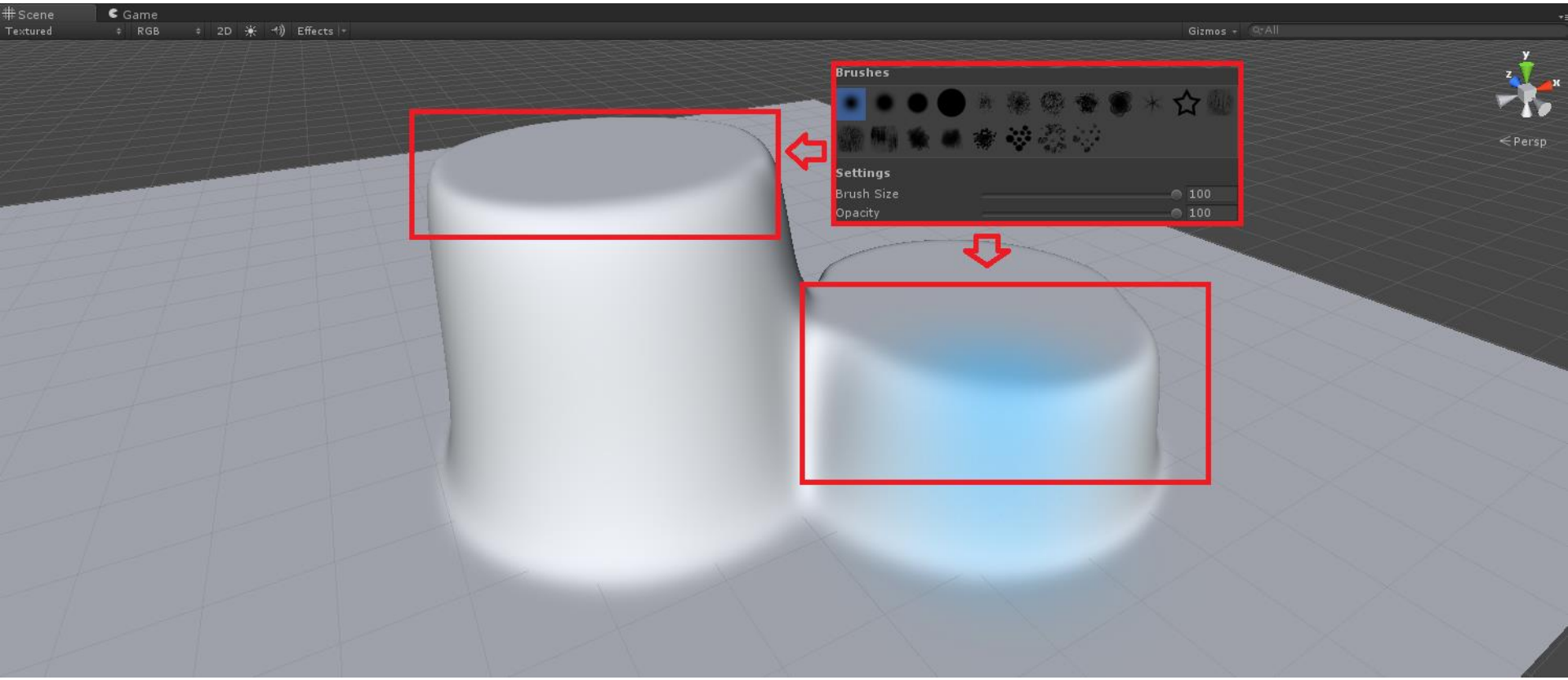
- 凹凸地形；使用筆刷繪製地形時，可以使地形產生突起效果，或是按住Shift鍵使地形有凹陷



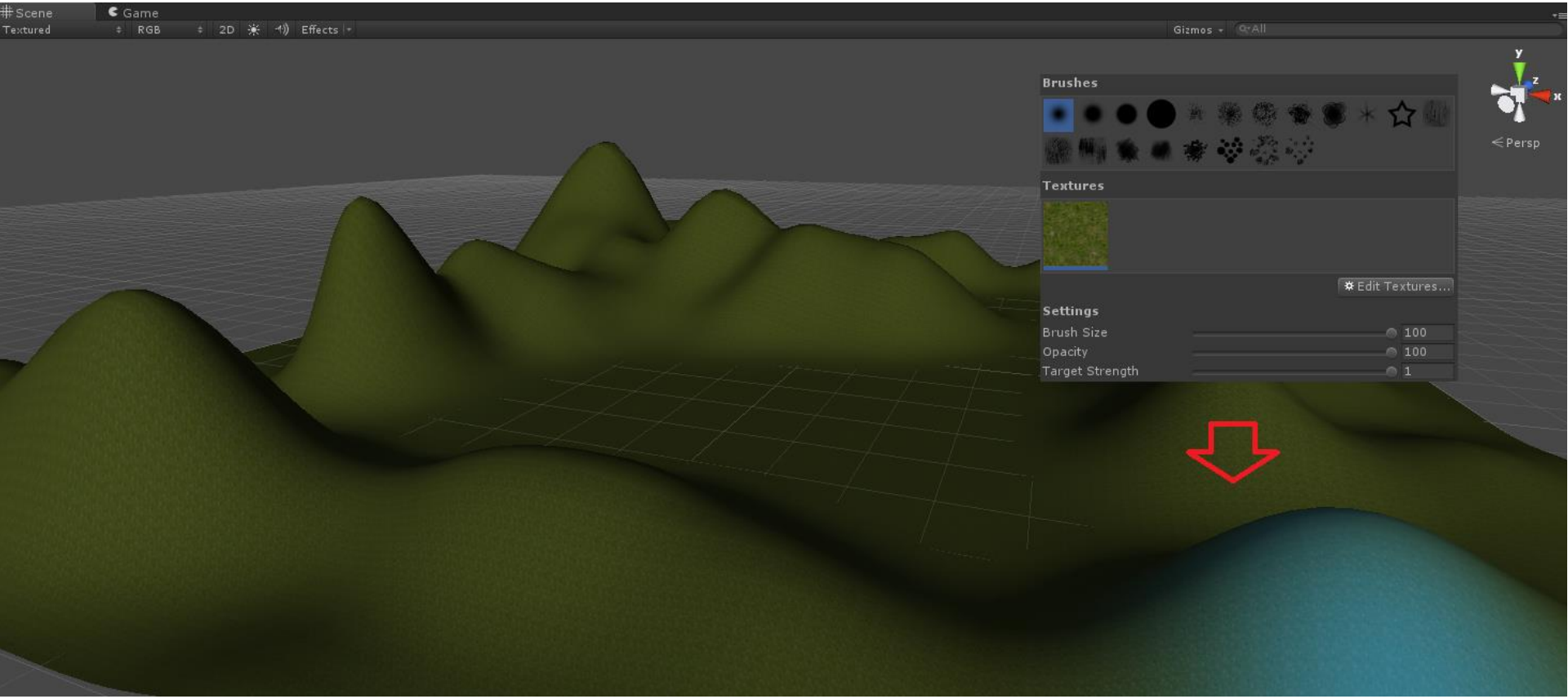
- 繪製高度：使用筆刷繪製地形時，可以使定地形同都高度為調整的Height參數。



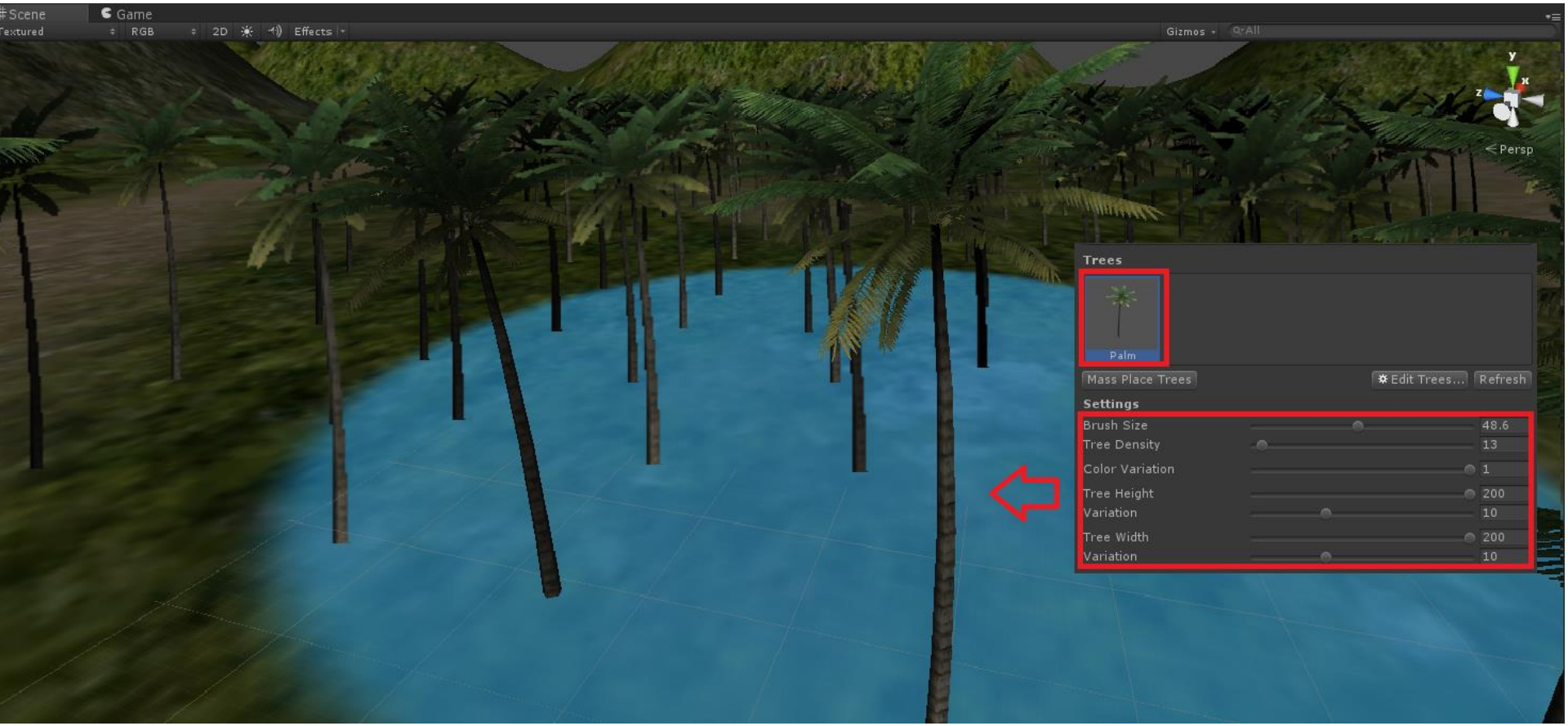
- 平滑地形：使用筆刷繪製地形時，會使地形較為平滑。



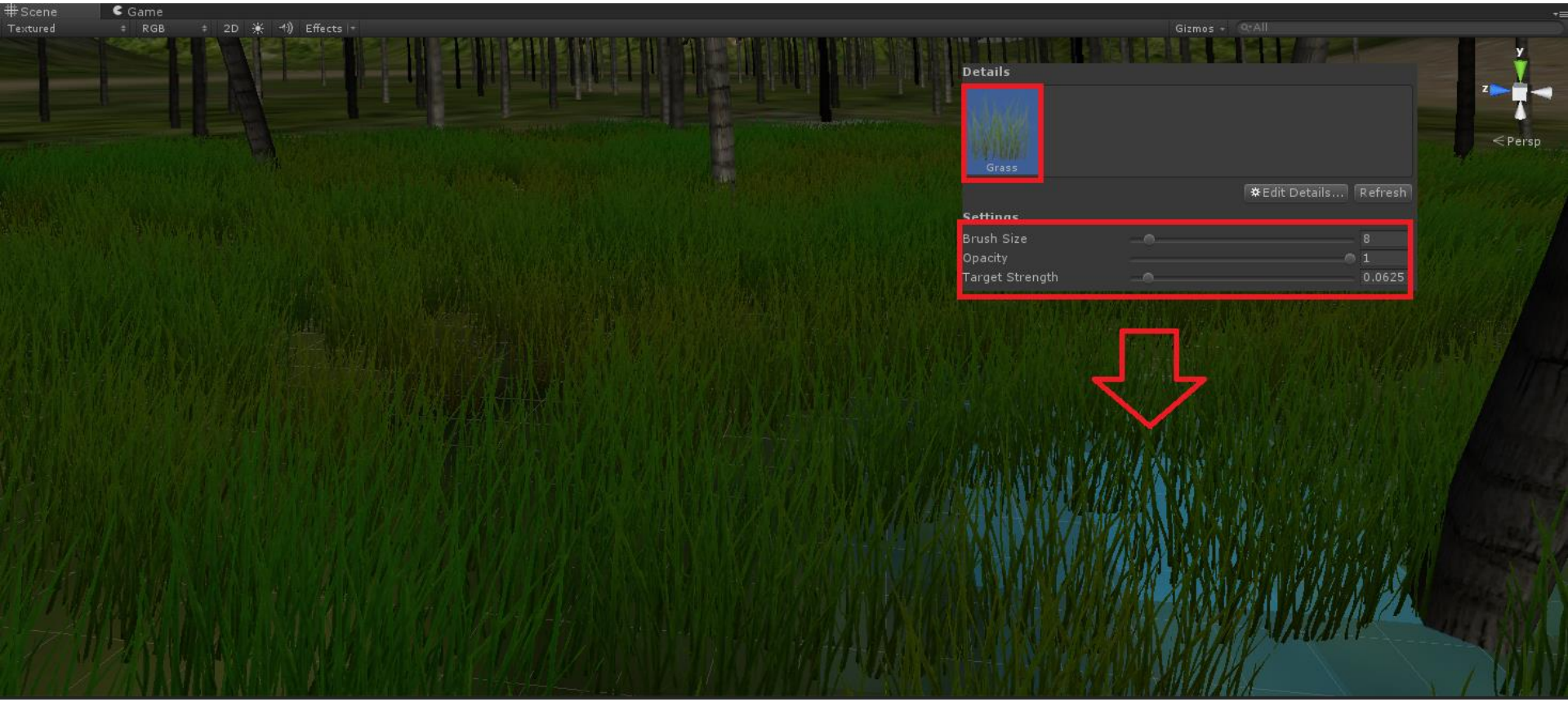
- 材質繪製：可在場景加入材質貼圖，此場景有植被、沙地等真實效果。



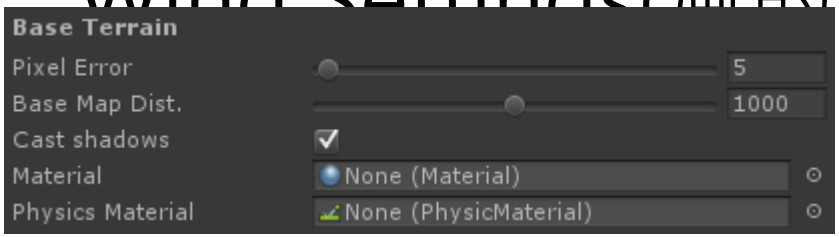
- 種植樹木：可在地形物件上種植樹木。

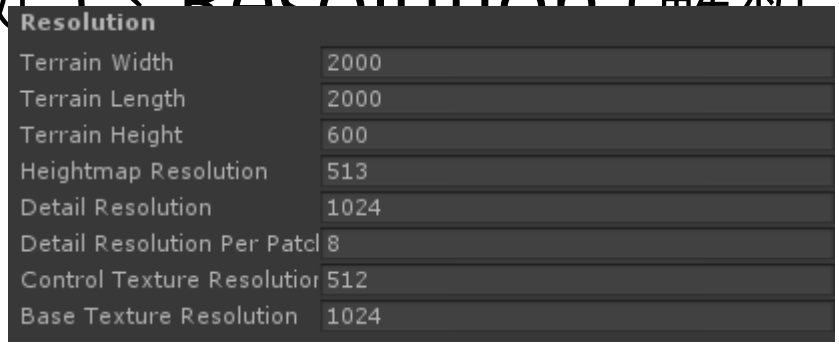


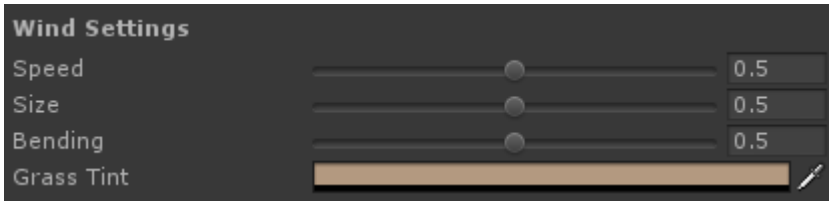
- 繪製細節：用於繪製地形細節，例如加上草叢、岩石等效果。

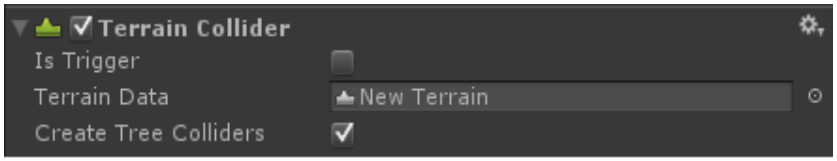


- 設定：地形物件的參數設定。其內參數主要分成4大項，分別為Base Terrain(基本地形)、Tree & Detail Object(樹木物件與細節物件)、Wind Settings(風的設定)、Resolution (解析





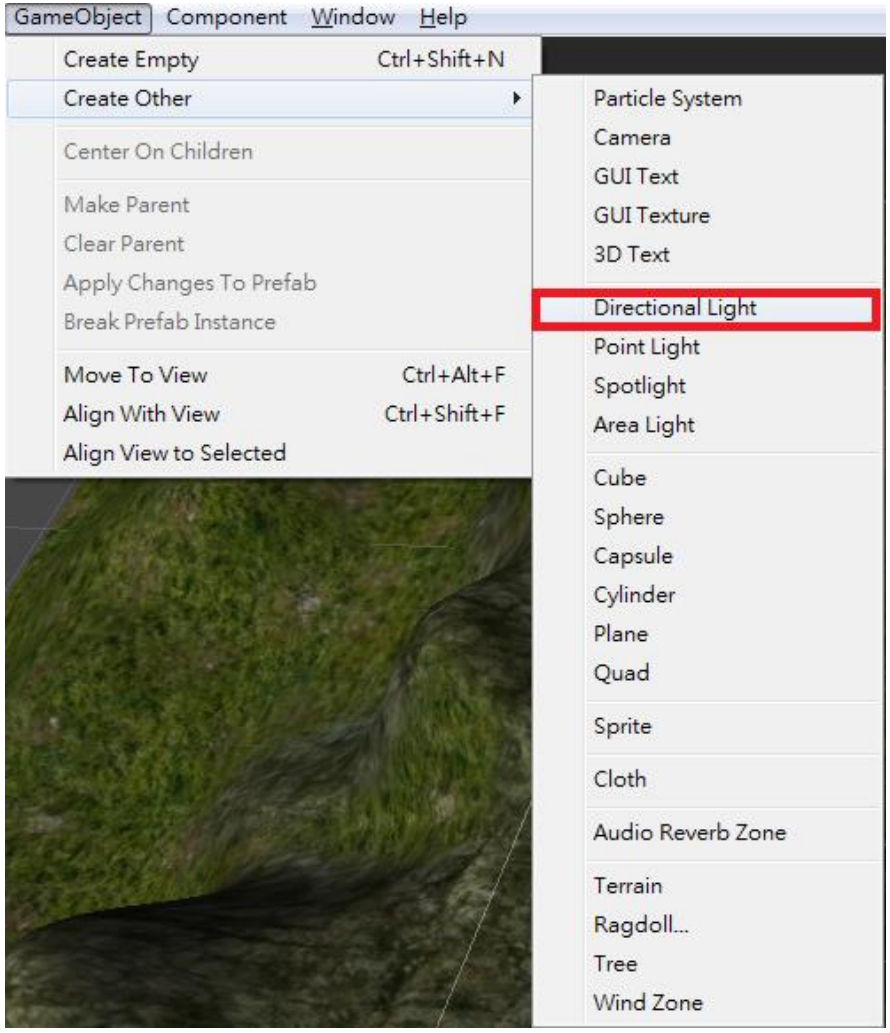


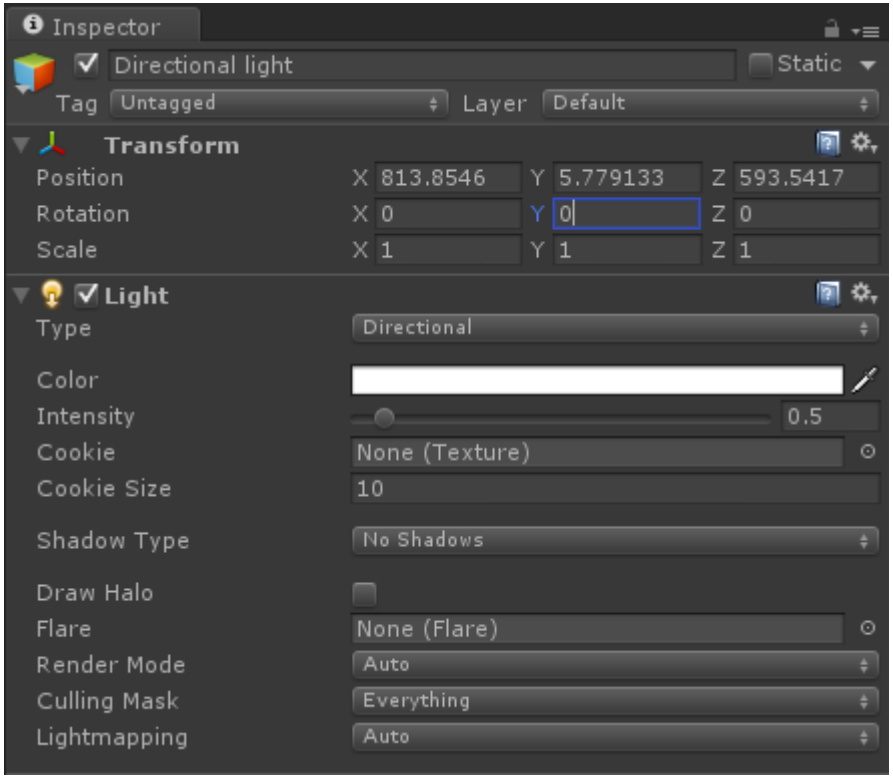


重點三:建立光源與第一人稱控制器

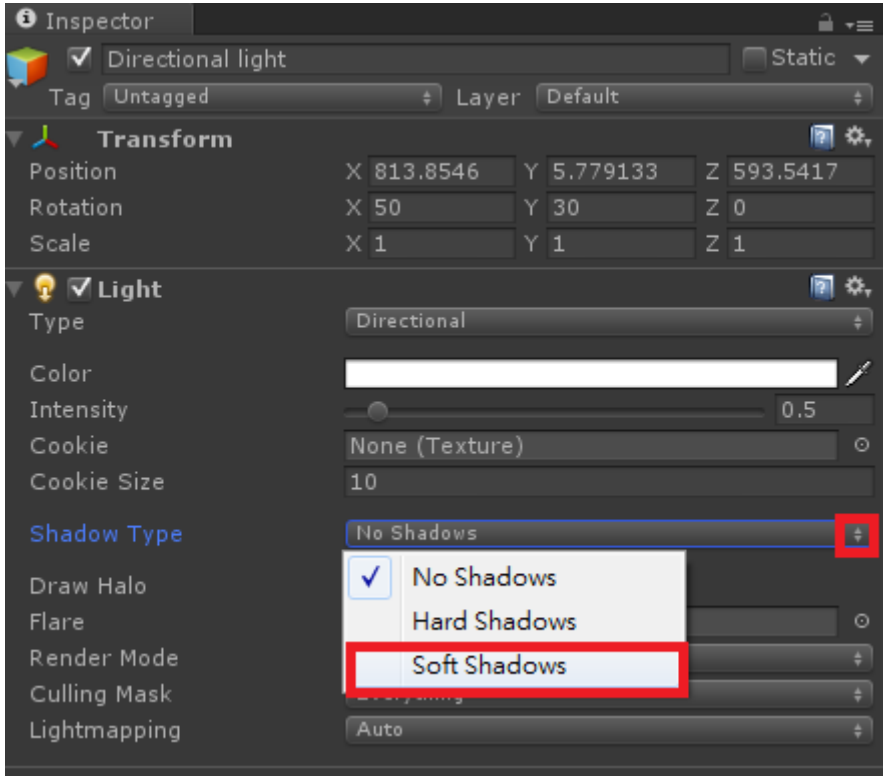
- 創建好場景後，便可以按在工具列的播放鈕，進入Game後，會發現怎麼場景是暗的，這是因為我們還沒有替場景打上燈光所造成的結果；與及進入Game後，我們也不能自由移動，只有一成不變的畫面，所以接下來要替我們的場景放入光源與人稱控制器。

如何建立光源

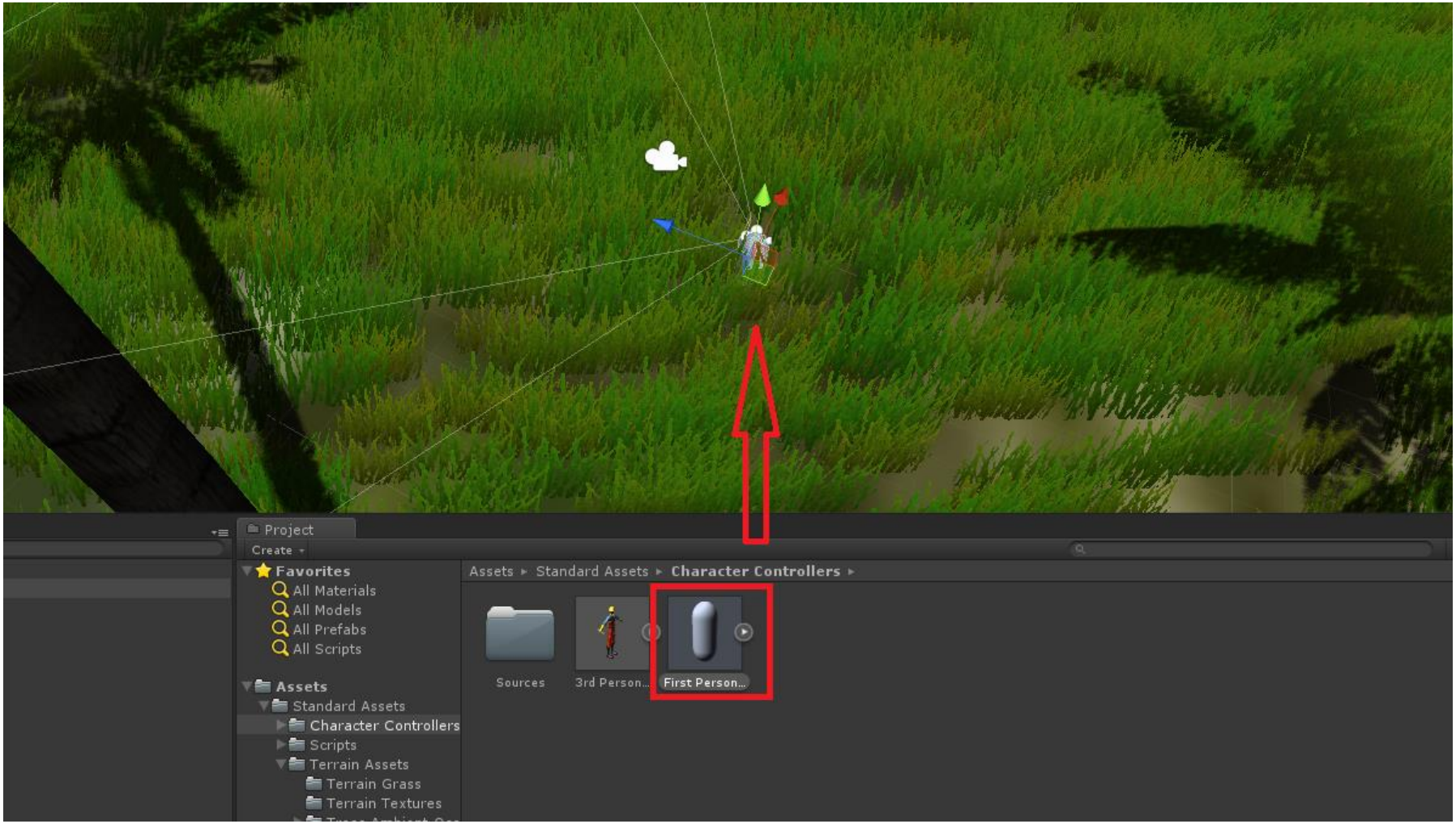

GameObject
Component
Window
Help
Create Empty
Ctrl+Shift+N
Create Other
Center On Children
Make Parent
Clear Parent
Apply Changes To Prefab
Break Prefab Instance
Move To View
Ctrl+Alt+F
Align With View
Ctrl+Shift+F
Align View to Selected
Particle System
Camera
GUI Text
GUI Texture
3D Text
Directional Light
Point Light
Spotlight
Area Light
Cube
Sphere
Capsule
Cylinder
Plane
Quad
Sprite
Cloth
Audio Reverb Zone
Terrain
Ragdoll...
Tree
Wind Zone


Inspector
Directional light
Static
Tag
Untagged
Layer
Default
Transform
Position
X 813.8546
Y 5.779133
Z 593.5417
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Light
Type
Directional
Color
Intensity
0.5
Cookie
None (Texture)
Cookie Size
10
Shadow Type
No Shadows
Draw Halo
Flare
None (Flare)
Render Mode
Auto
Culling Mask
Everything
Lightmapping
Auto

如何新增影子效果


Inspector
Directional light
Static
Tag
Untagged
Layer
Default
Transform
Position
X 813.8546
Y 5.779133
Z 593.5417
Rotation
X 50
Y 30
Z 0
Scale
X 1
Y 1
Z 1
Light
Type
Directional
Color
Intensity
0.5
Cookie
None (Texture)
Cookie Size
10
Shadow Type
No Shadows
Draw Halo
Flare
Render Mode
Culling Mask
Lightmapping
Auto
No Shadows
Hard Shadows
Soft Shadows

如何建立第 一人稱控制器



單元四 場景設計

遊戲場景中不同風貌的設置

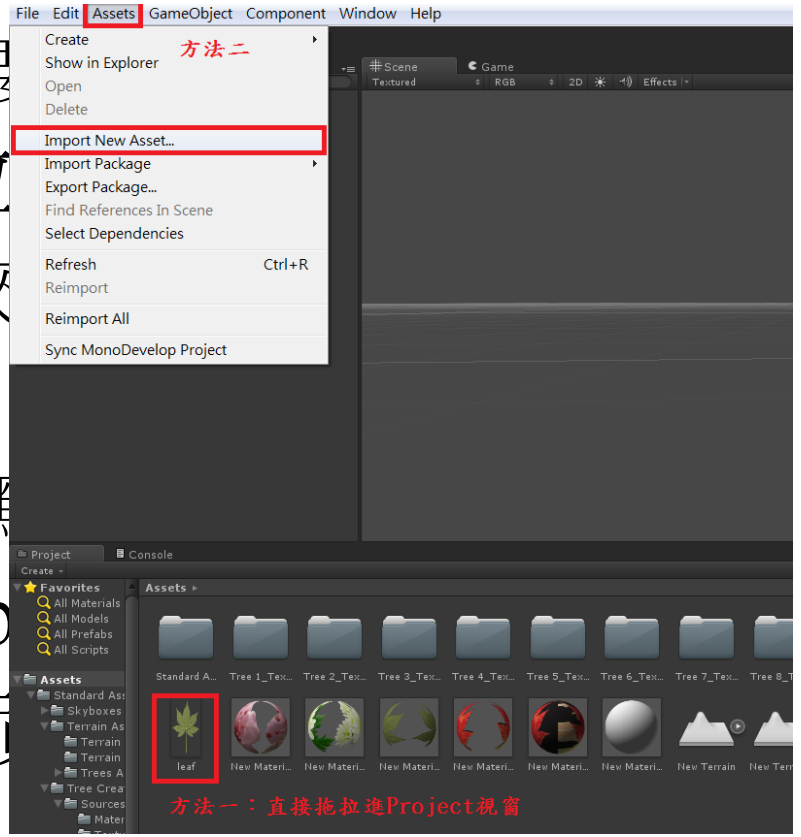
學習重點

- 外部圖片資源的匯入
- 建立地形上的樹木及風
- 建立天空盒改變天空的樣式

外部圖片資源的匯入

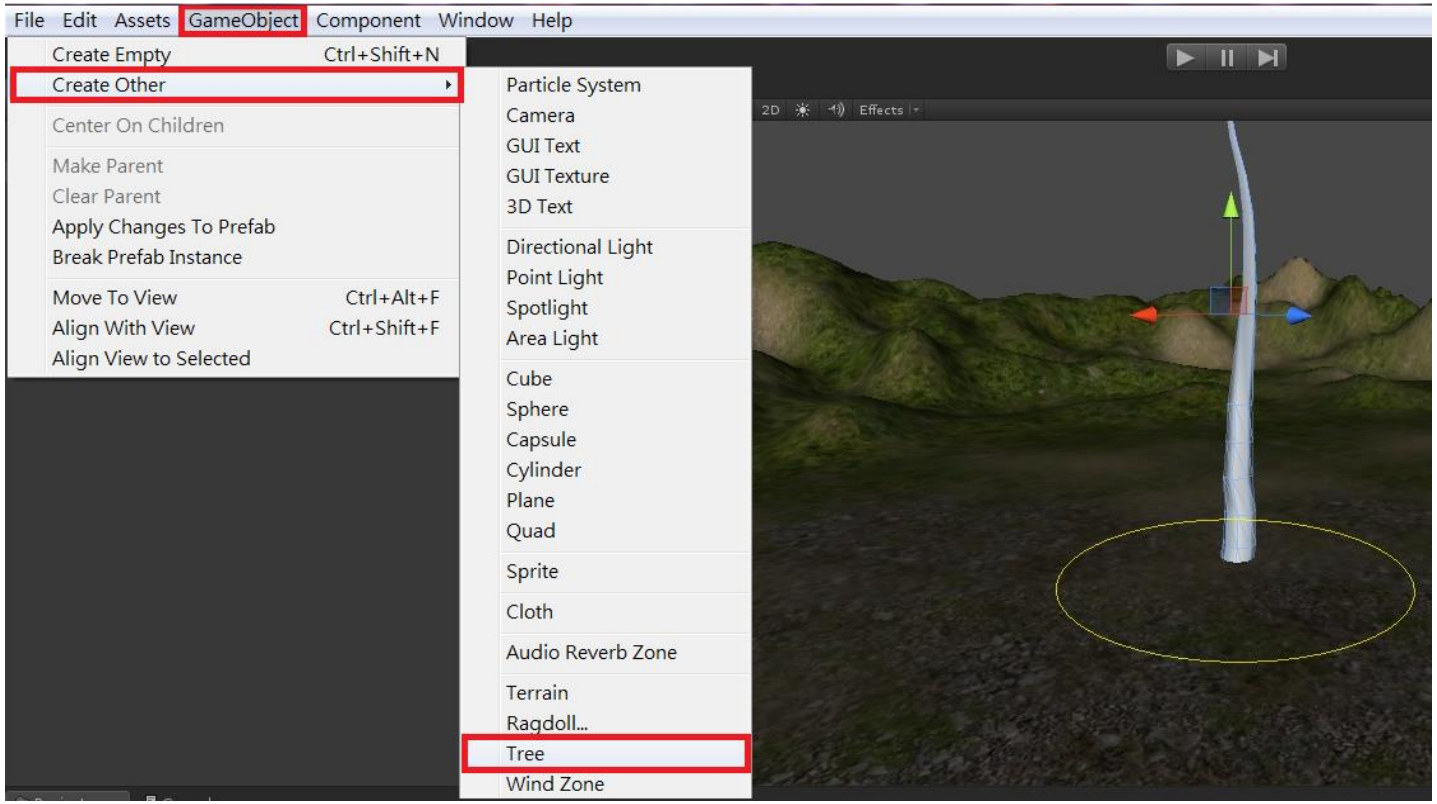
– 第一種匯入方式是將想要
圖片資源用拖拉的方式拉
Project視窗的Assets文件夾

– 第二種匯入方式是直接點
左上方的Assets在點選Imp
New Asset … 來新增圖片資

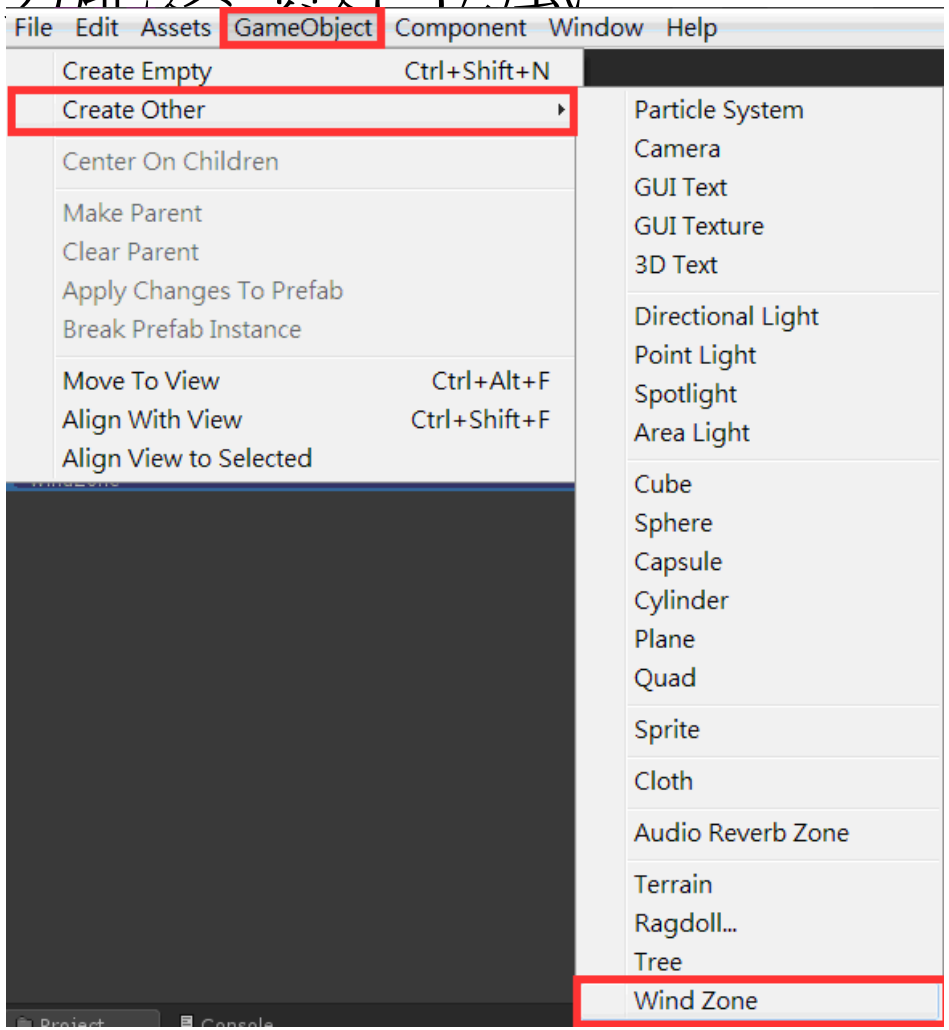


建立地形上的樹木及風

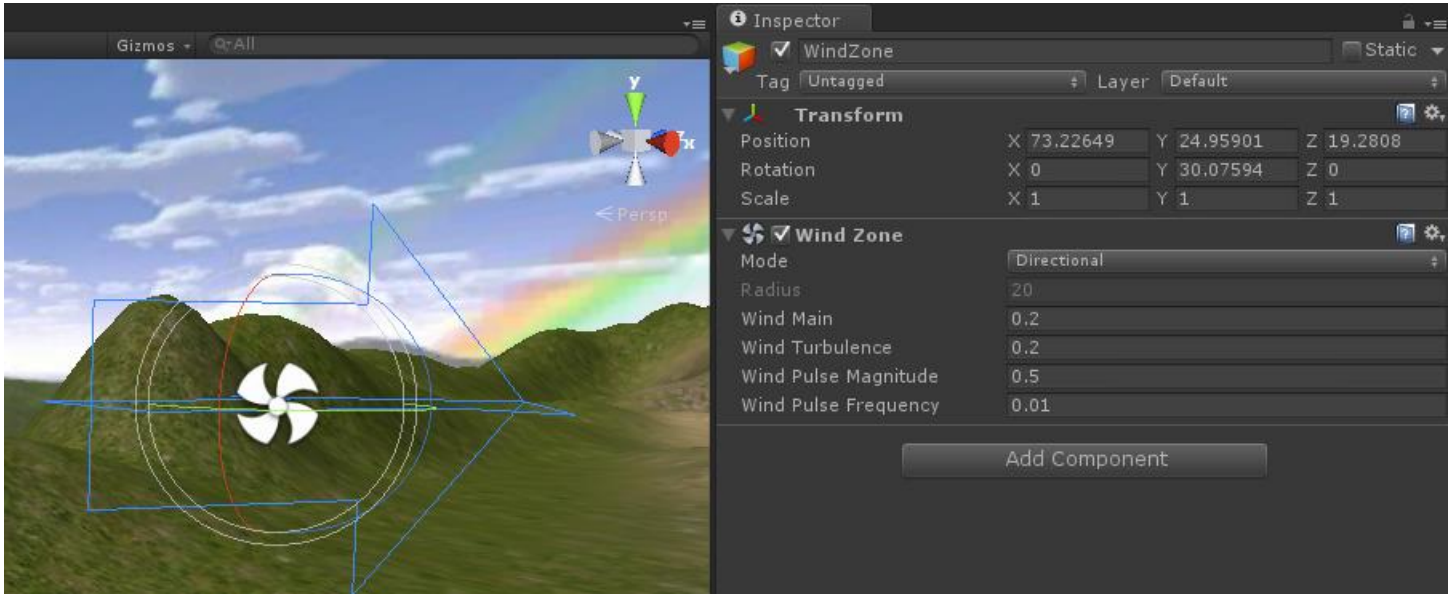
- 從GameObject裡點選Create Other再點選 Tree 的功能選項建立樹。



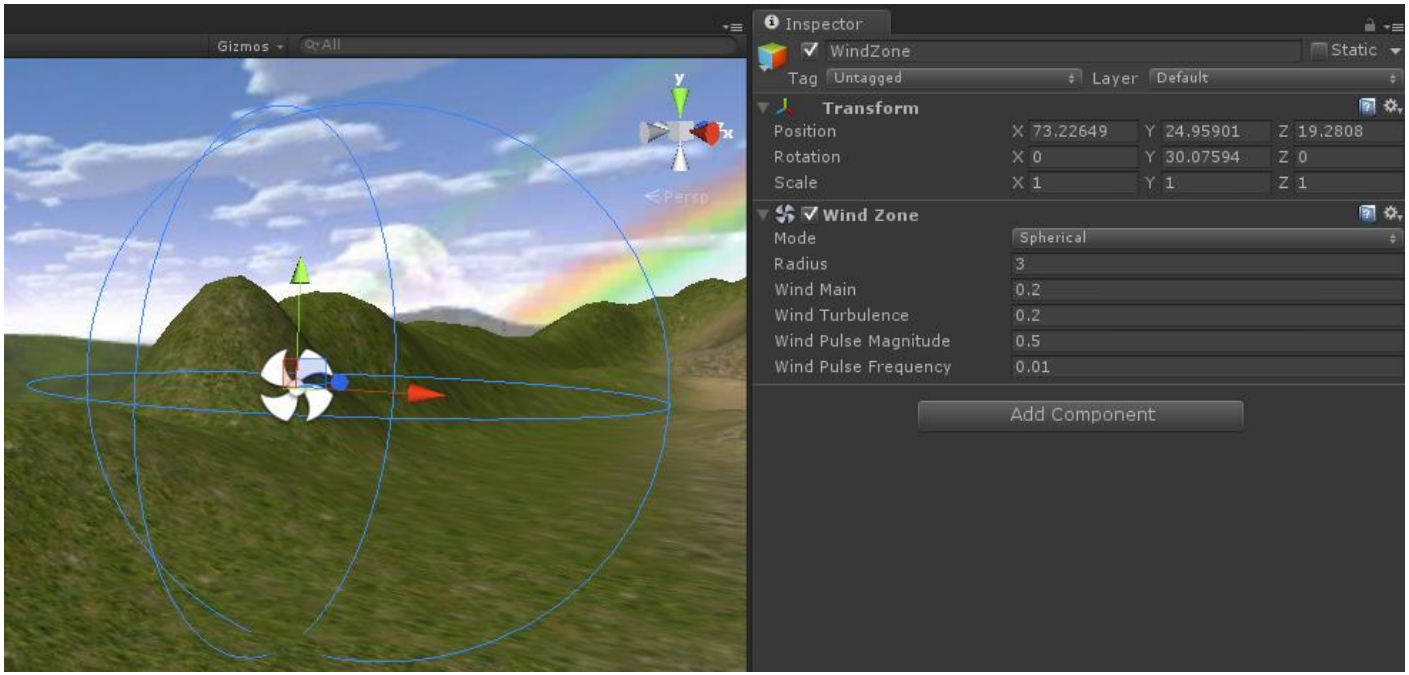
- 從GameObject裡點選Create Other再點選Wind Zone的功能選項建立風。



方向風

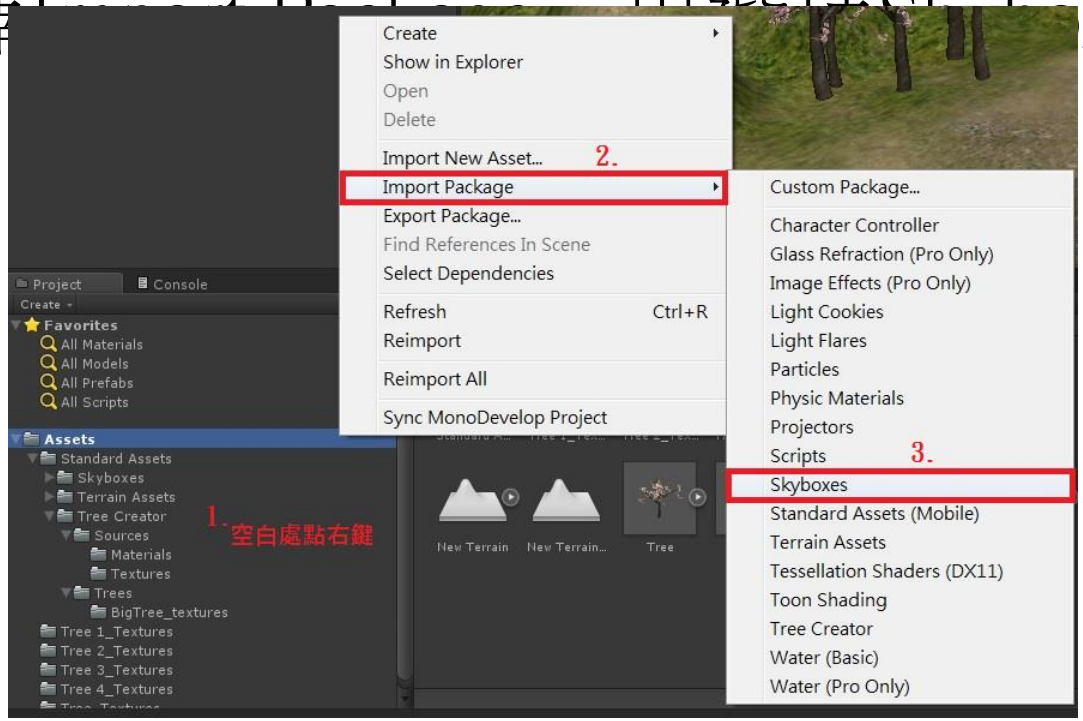

Gizmos
All
y
x
Persp
Inspector
WindZone
Static
Tag
Untagged
Layer
Default
Transform
Position
X 73.22649
Y 24.95901
Z 19.2808
Rotation
X 0
Y 30.07594
Z 0
Scale
X 1
Y 1
Z 1
Wind Zone
Mode
Directional
Radius
20
Wind Main
0.2
Wind Turbulence
0.2
Wind Pulse Magnitude
0.5
Wind Pulse Frequency
0.01
Add Component

區域風

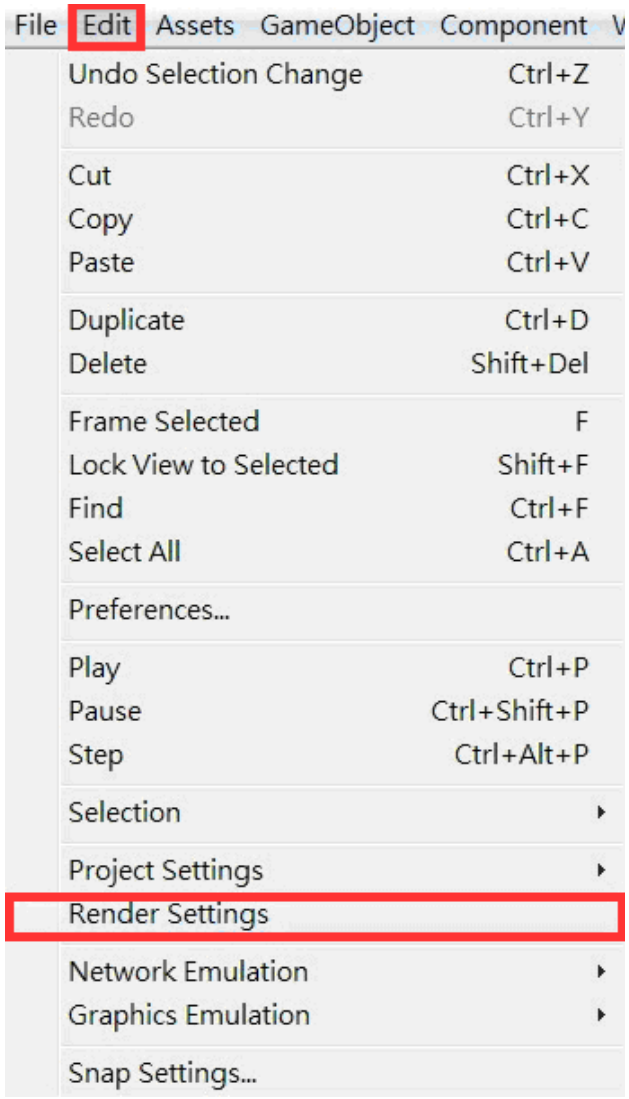

Gizmos
All
Persp
Inspector
WindZone
Static
Tag
Untagged
Layer
Default
Transform
Position
X 73.22649
Y 24.95901
Z 19.2808
Rotation
X 0
Y 30.07594
Z 0
Scale
X 1
Y 1
Z 1
Wind Zone
Mode
Spherical
Radius
3
Wind Main
0.2
Wind Turbulence
0.2
Wind Pulse Magnitude
0.5
Wind Pulse Frequency
0.01
Add Component

建立天空盒改變天空的樣式

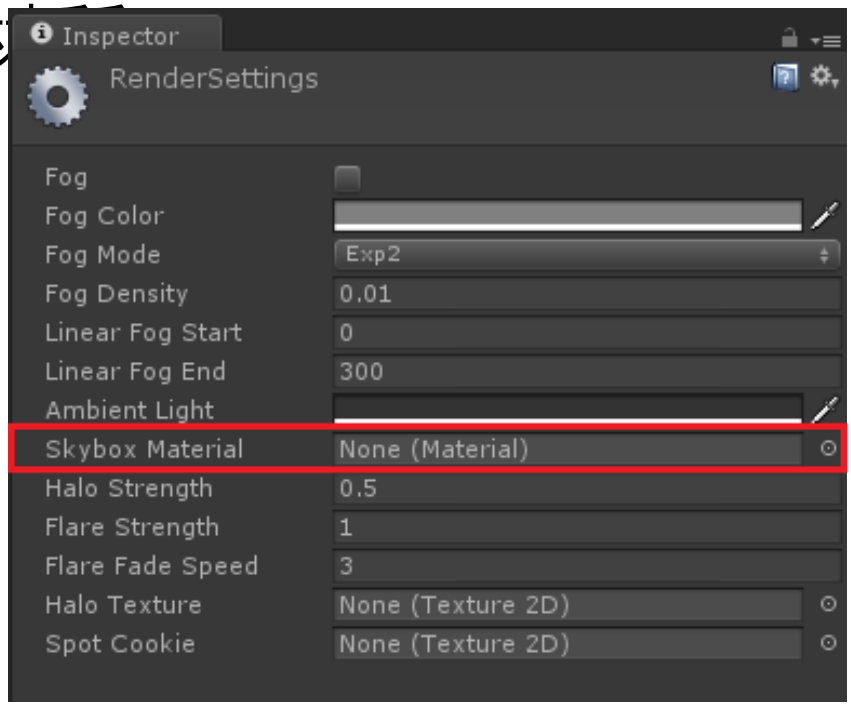
- 在Project視窗的Assets文件夾匯入Shyboxes來建立內建的天空盒，然後在Project視窗的Assets文件夾中按滑鼠右鍵叫出功能選項，選擇I[illegible]，再選擇S[illegible]xes。



- 從Edit裡點選Render Settings的功能選項，如下圖示。

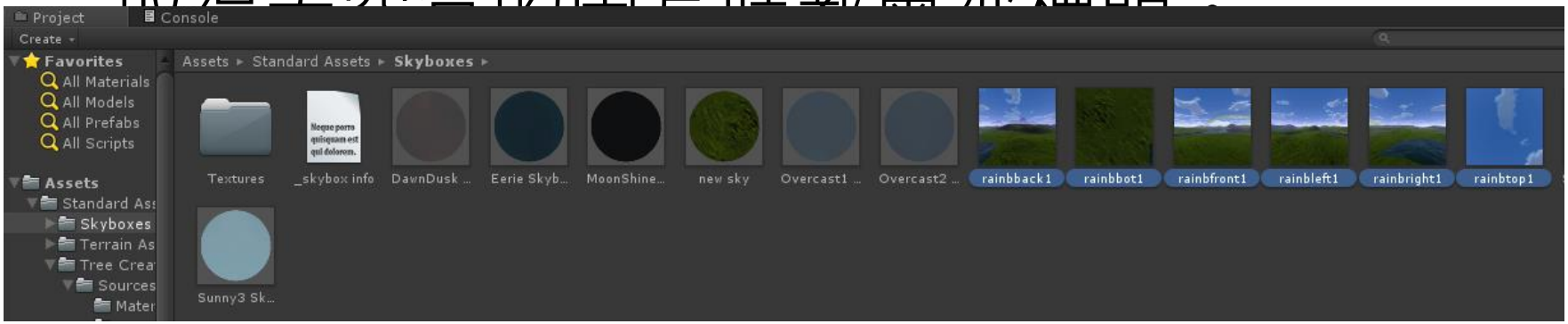


- 在Inspector視窗裡可以看到Skybox Material，在右邊的小圓點就可選擇內建有的天空材質

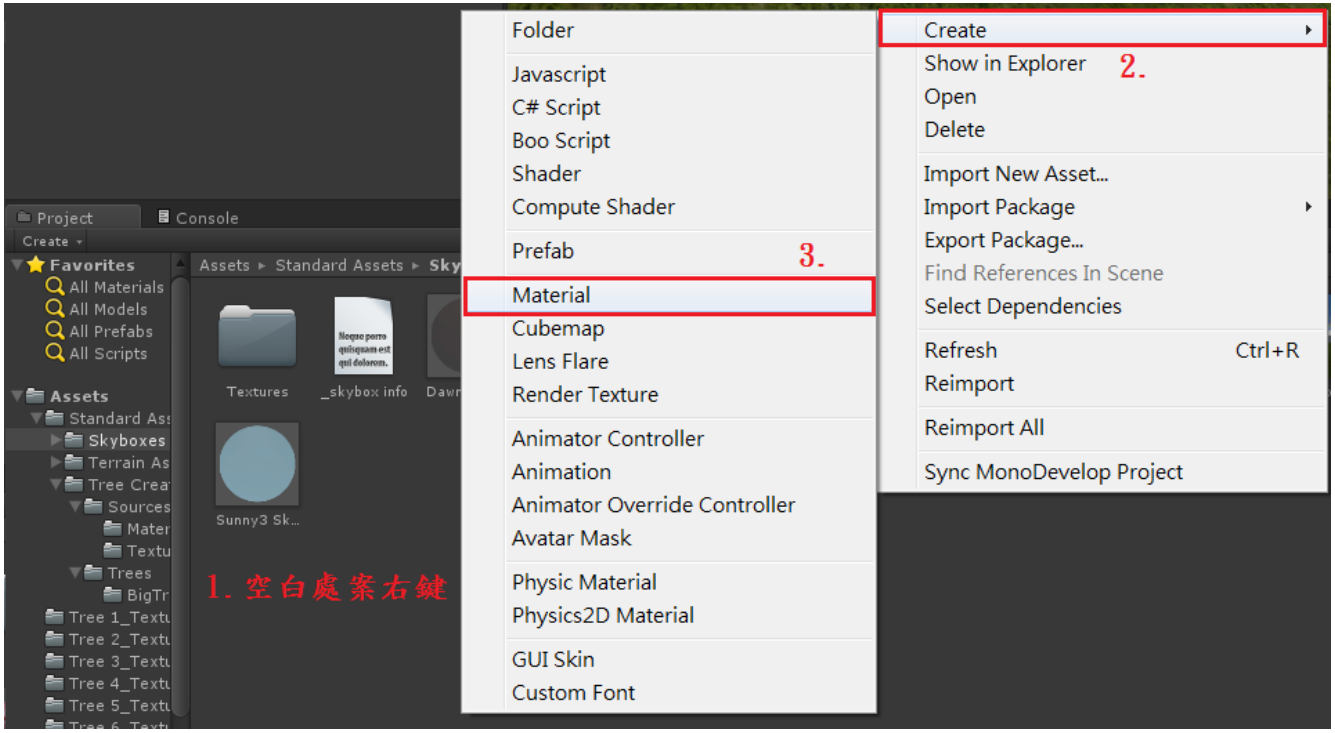


自製新的天空盒改變天空的樣式

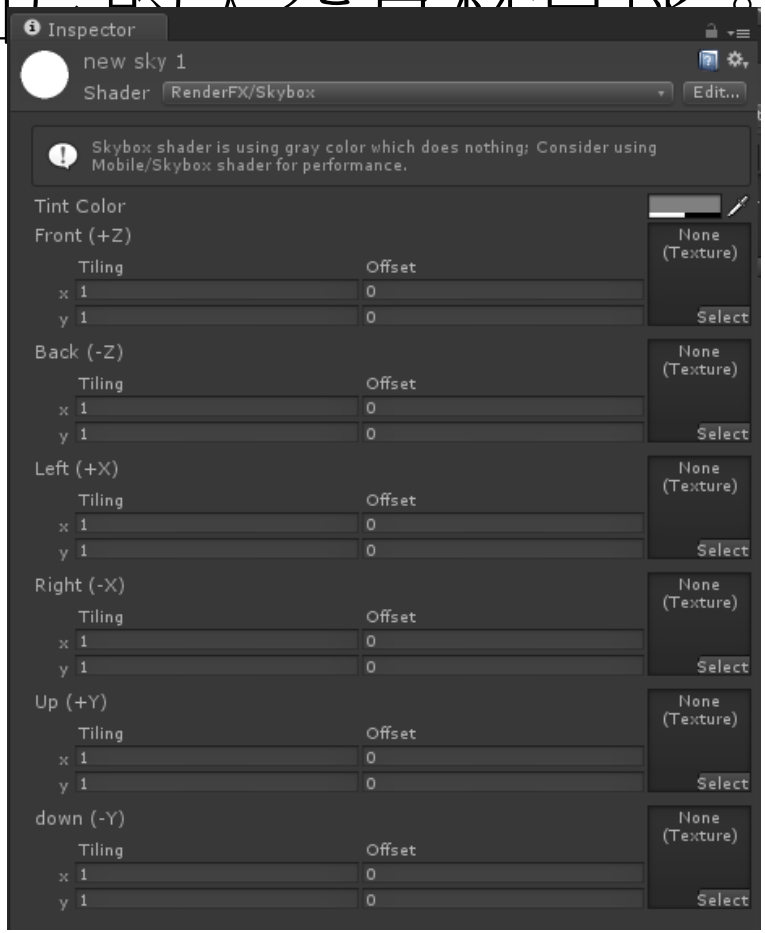
- 在Project視窗的Assets文件夾用拖拉的方式匯入想要的天空的六張圖片紋理，分為前、後、左、右、上、下，這部分一般在取得天空盒的圖片時都會被標明。



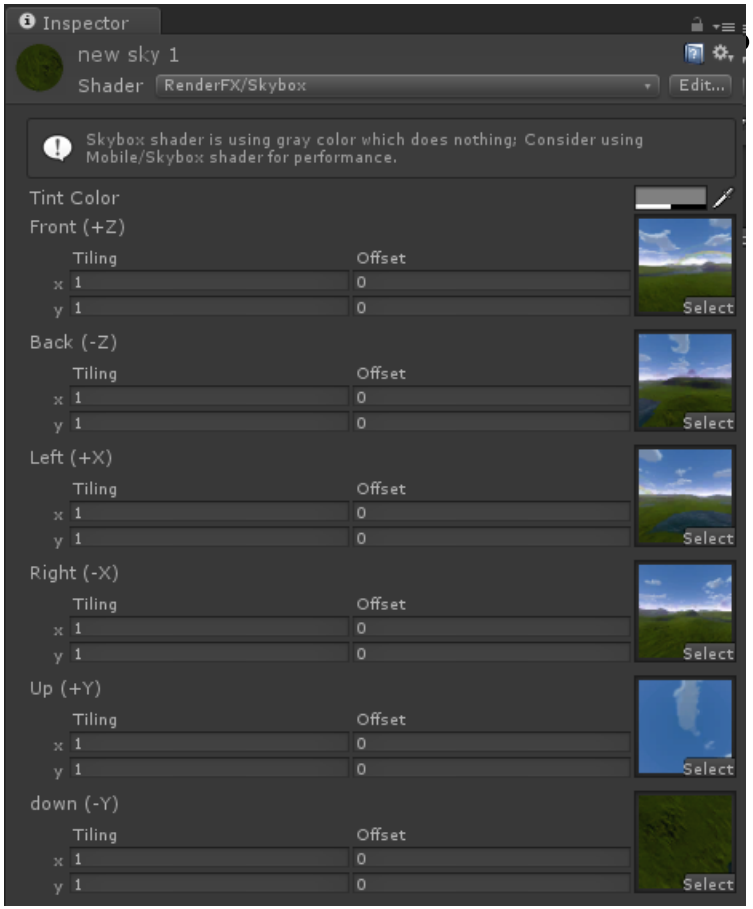
- 了解之後在Project視窗的Assets文件夾新增一個材質球，如下圖示。



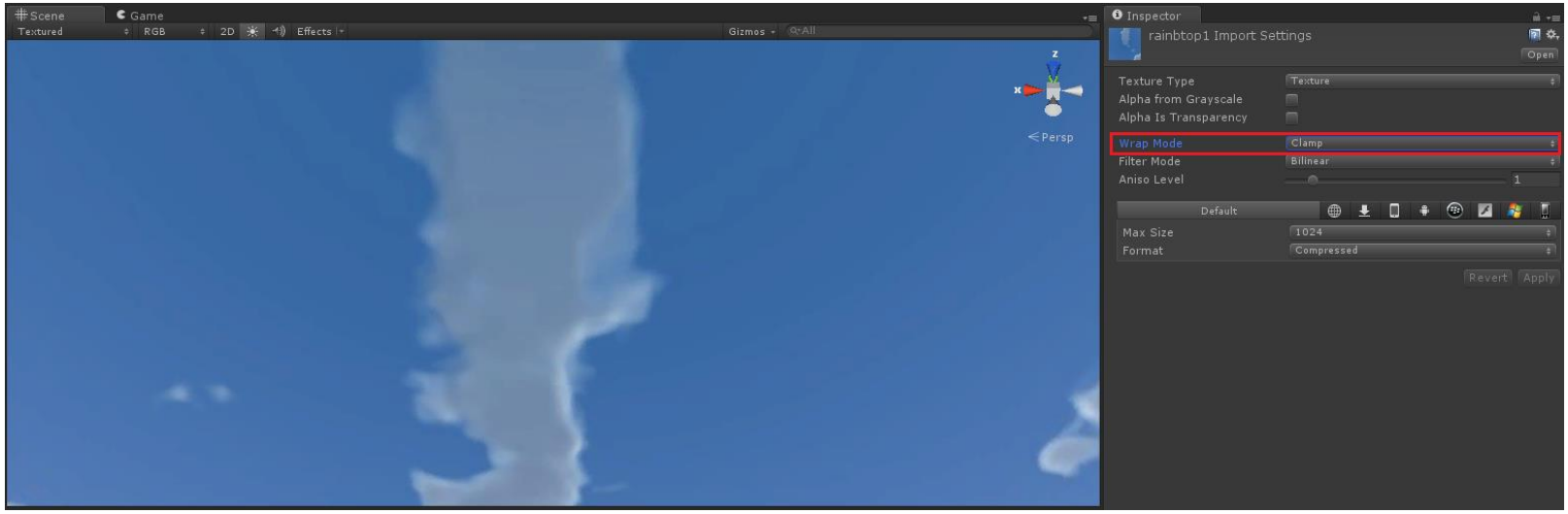
- 在Inspector視窗的Shower點選RenderFX再點選Skybox的功能選項，Inspector視窗就會變成六張圖片的天空盒材質球。



- 點選Inspector視窗裡的select將相對應位置的貼圖貼上也就是區分出前、後、左、右、上、



- 選擇天空盒材質所用的紋理，在Inspector視窗中，將該紋理的Repeat(重複)設置為Clamp(截斷)且按一下Apply。



單元五 特效

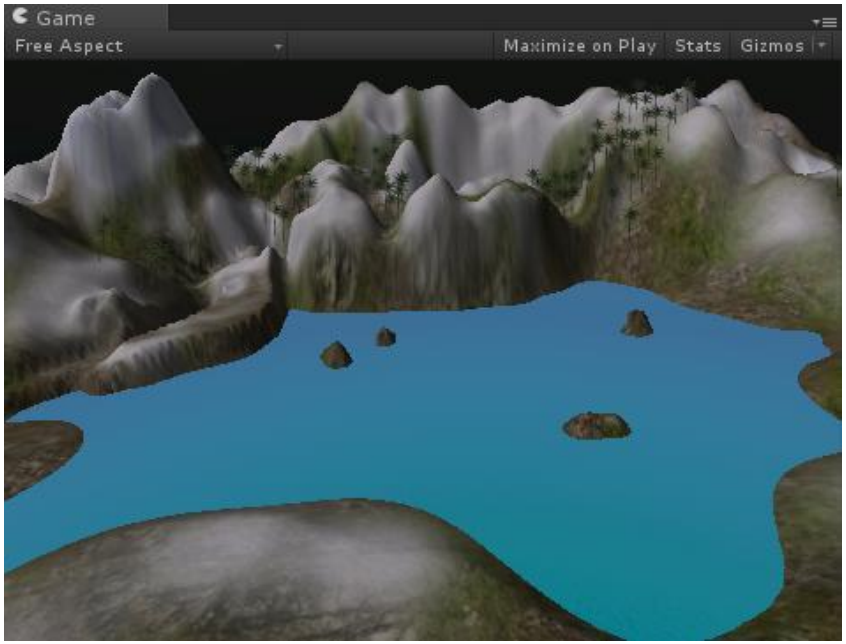
遊戲場景中水特效及粒子動態效果

學習重點

- 在場景中利用水資源包建立水特效
- 在場景中建立動態粒子特效

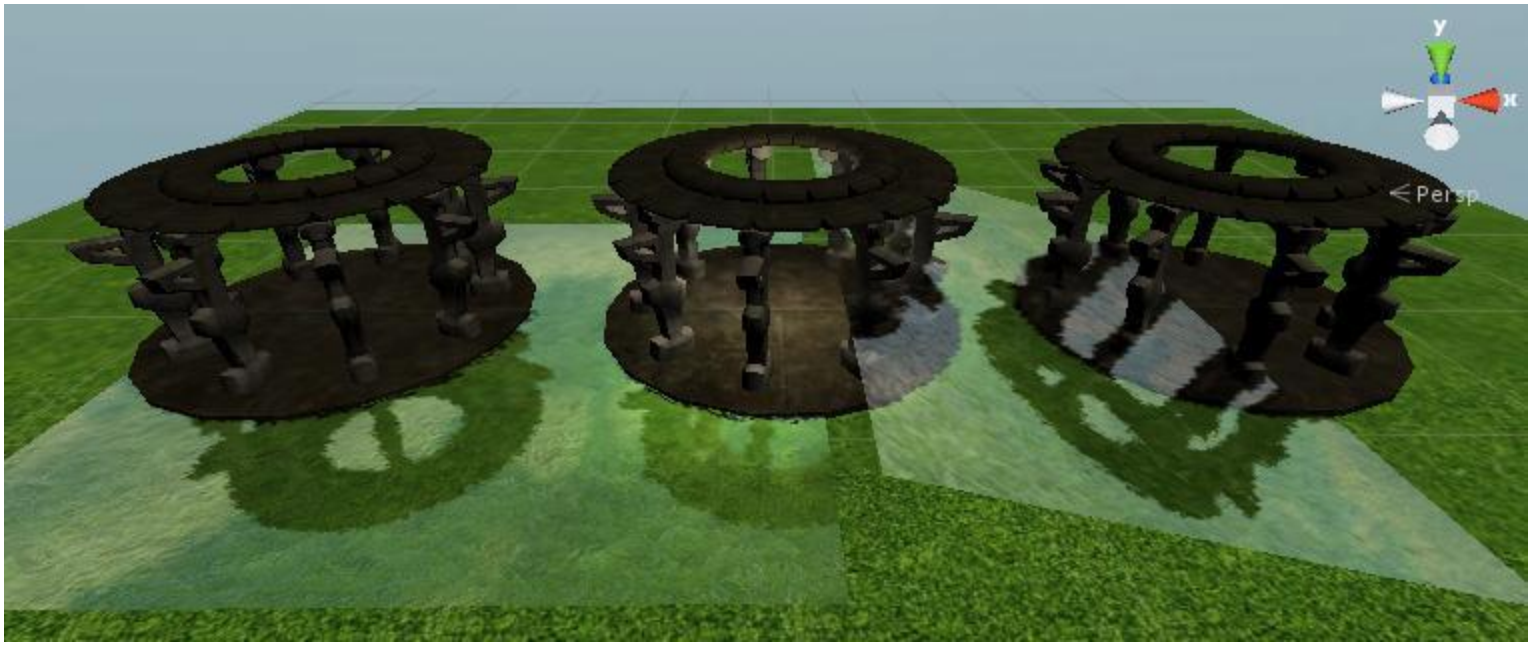
在場景中利用水資源包建立水特效

Water(Basic)　　Water(Pro Only)

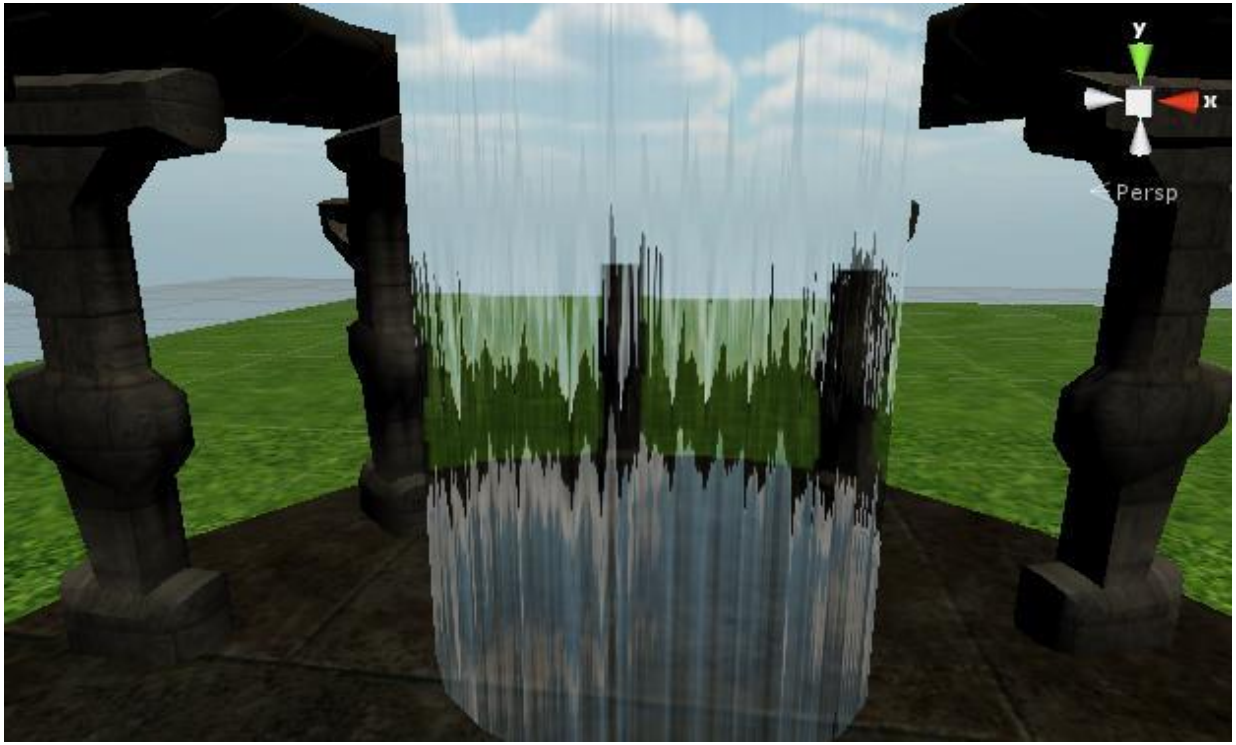


- 能對遊戲場景中的天空與物體進行反射或折射運算並產生水波效果。

- 可以同時在場景中放置多個水的區域，不同的水區域會因為位置高低與旋轉角度的不同而反射出不同的倒影。

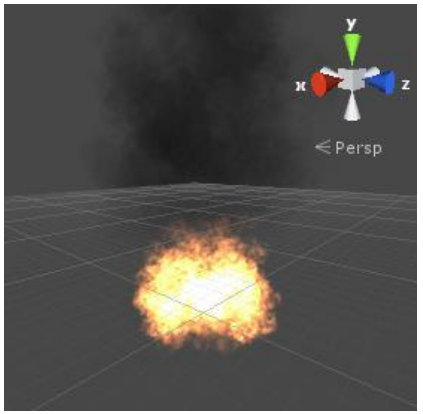


- 水區域的形狀有多樣的變化。



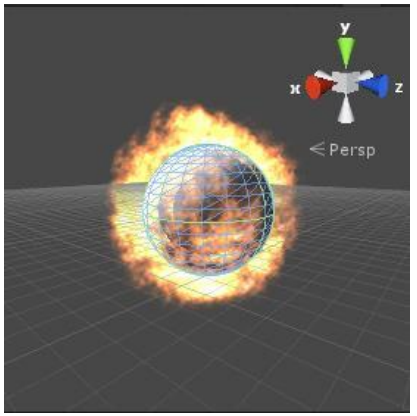
在場景中建立動態粒子特效

- 利用二維的圖片經由不斷重複的生成，進而在三維的空間中產生的動態效果。



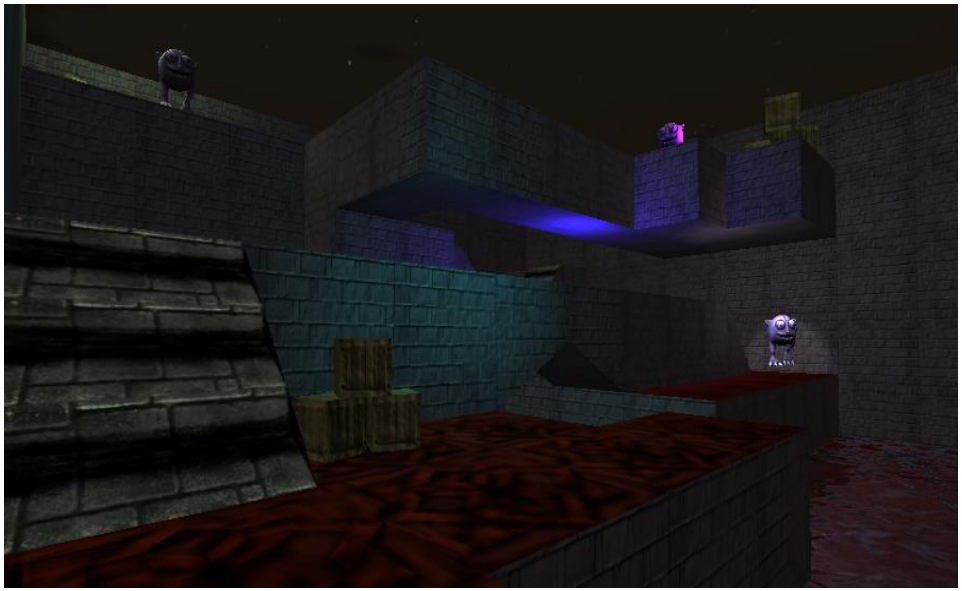
- 粒子系統

Ellipsoed Particle(橢球粒子發射器)
Mesh Paticle Emitter(網格粒子發射器)
Particle Animator(粒子動畫器)
Particle Render(粒子渲染器)
World Particle Collider(粒子碰撞器)



單元六 打光

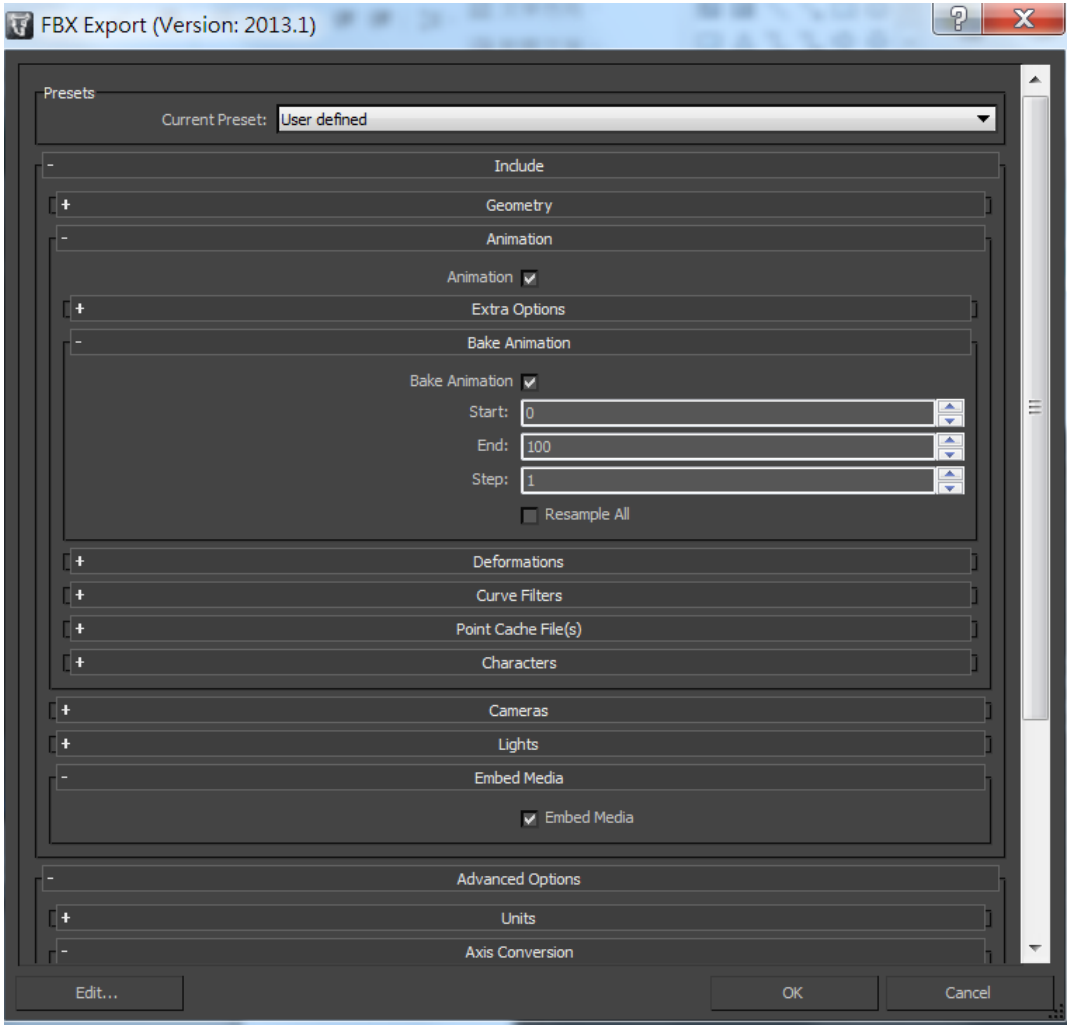
遊戲模型的匯入與場景打光



學習重點

- 外部模型匯入Unity-以3ds Max為例
- 替遊戲場景加上燈光效果

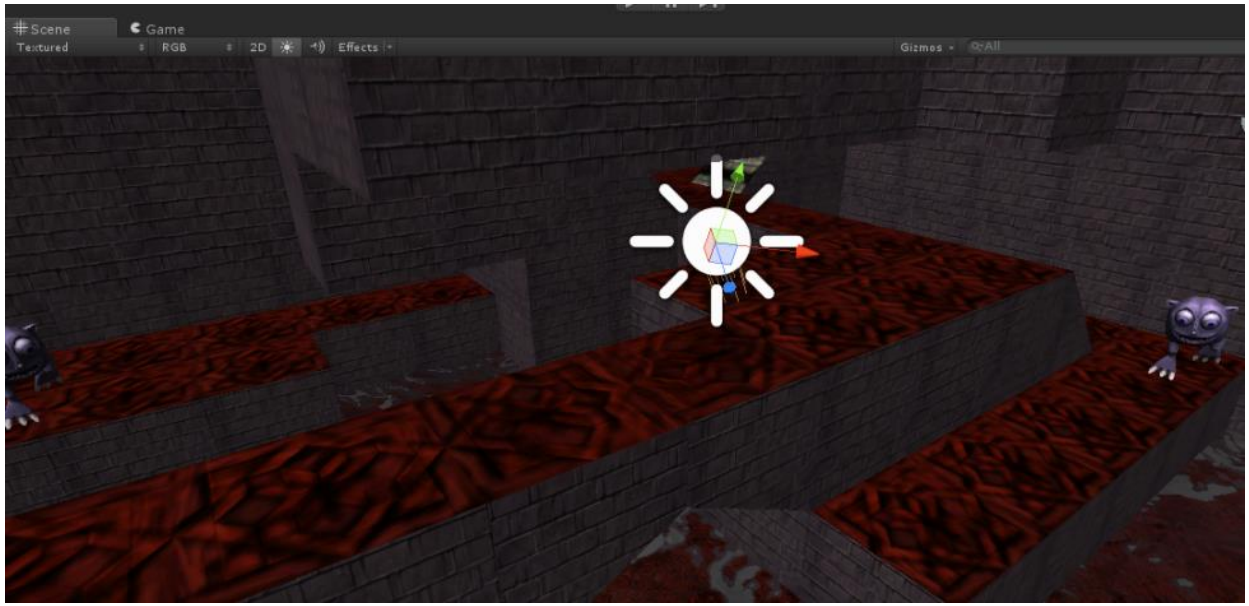
外部模型匯入Unity-以3ds Max為例



替遊戲場景加上燈光效果

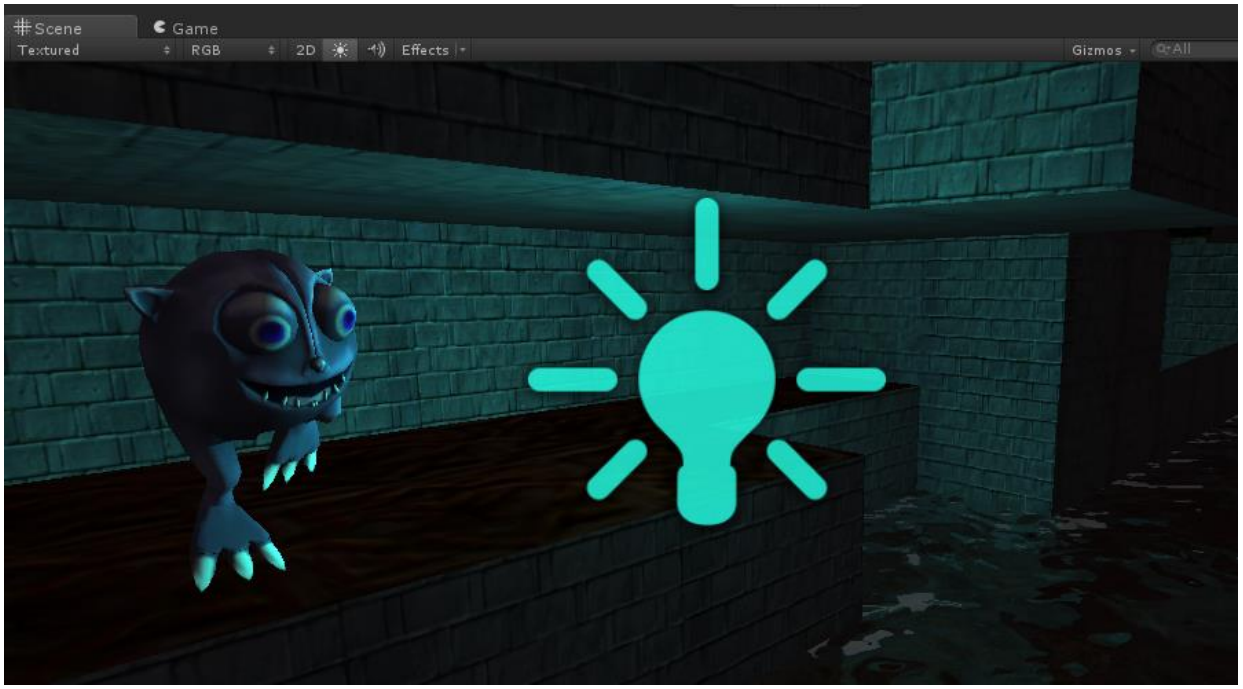
- Directional Light(方向光源)
- Point Light(點光源)
- Spotlight(聚光燈)
- Area Light(區域光)

Directional Light(方向光源)



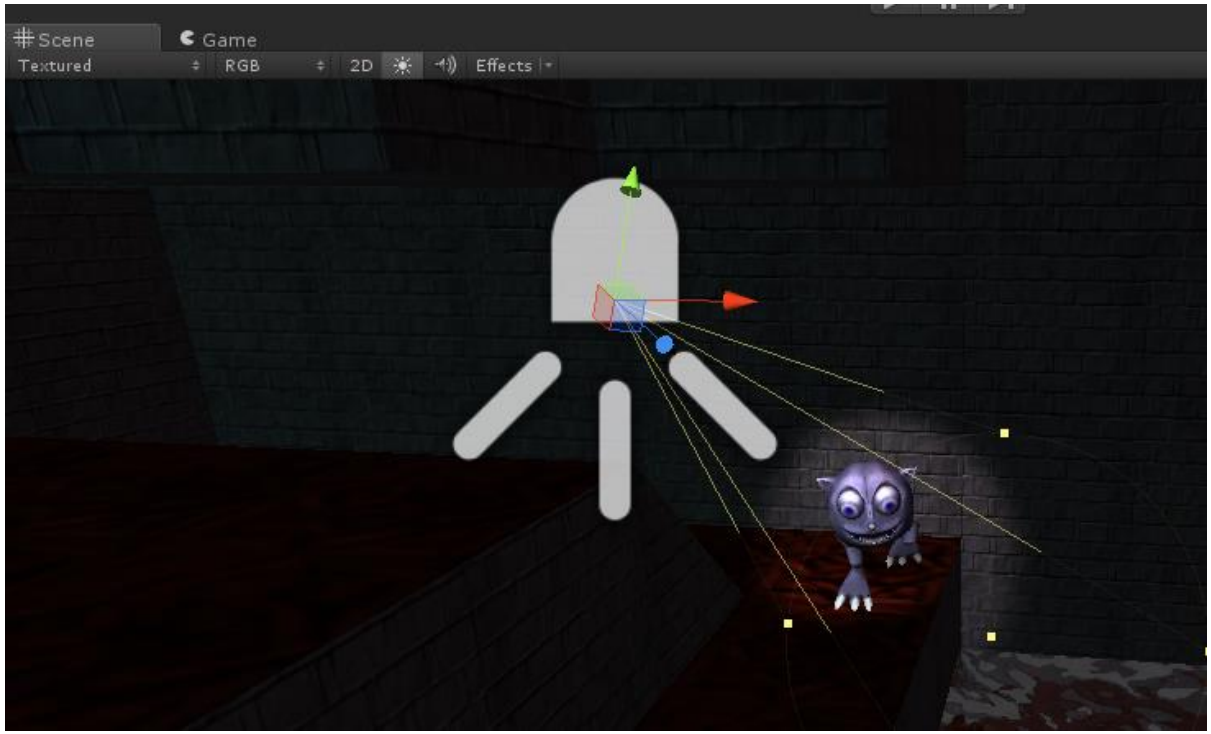
該類型光源可以被放置在無窮遠的地方，能影響場景中的所有物件，類似於自然界中日光的照明效果。

Point Light(點光源)



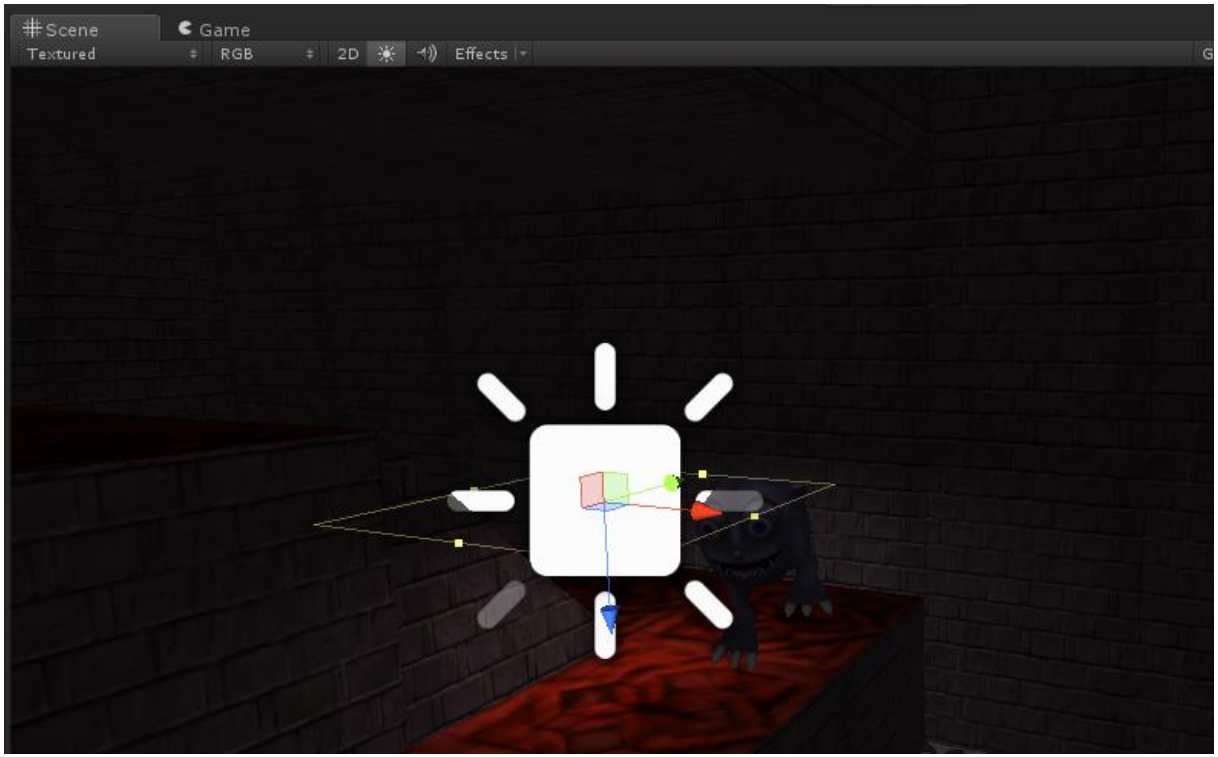
它是一個位置向四面八方，影響其範圍內的所有物件，類似燈泡的照明效果。

Spotlight(聚光燈)



該類型的燈光從一點發出，在一個方向按照一個錐形的範圍照射，處於錐形區域內的對象會受到光線照射，類似射燈的照明效果。

Area Light(區域光)



該類型光源無法應用於即時光照，僅適用於光照貼圖烘焙。

單元七 粒子特效

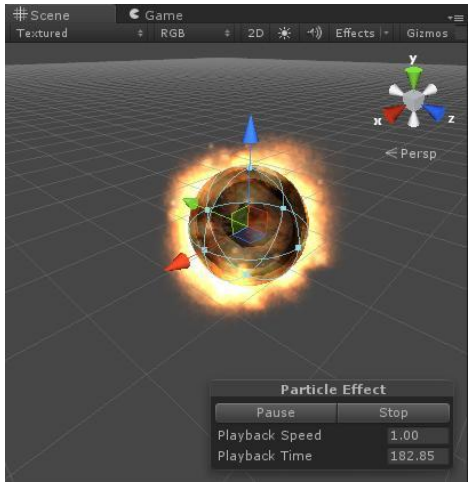
探討Shuriken粒子系統的特效應用

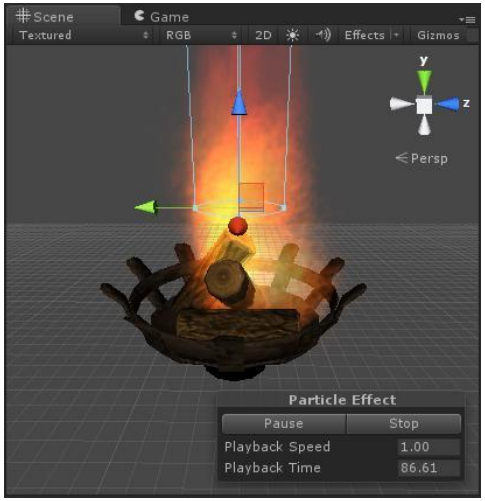
學習重點

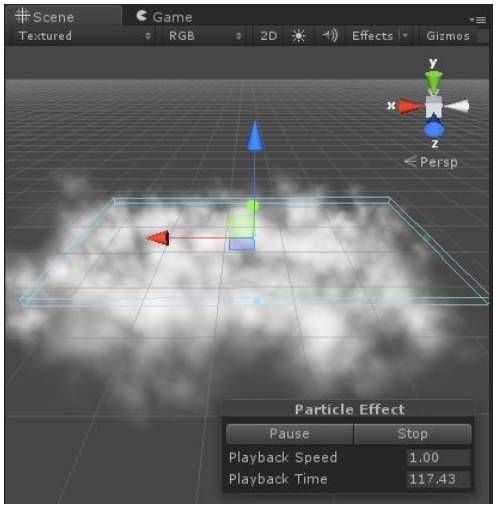
- 介紹Shuriken粒子系統
- Unity音頻的匯入與使用

介紹Shuriken粒子系統

- 利用二維的圖片經由不斷重複的生成，進而在三維的空間中產生的動態效果。

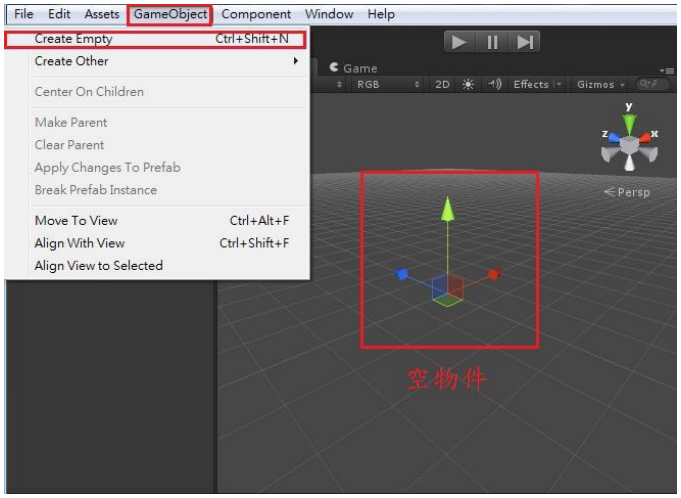


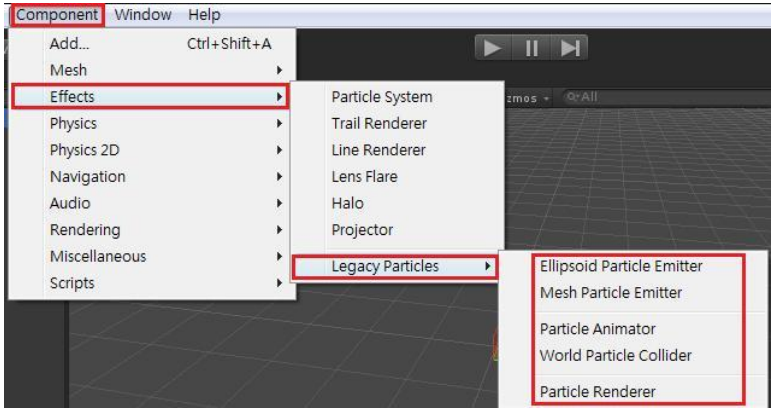


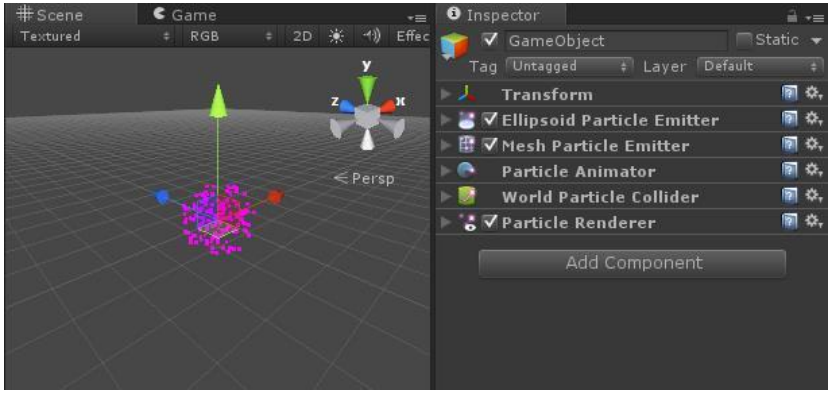


比較粒子系統與Shuriken粒子系統添加方式

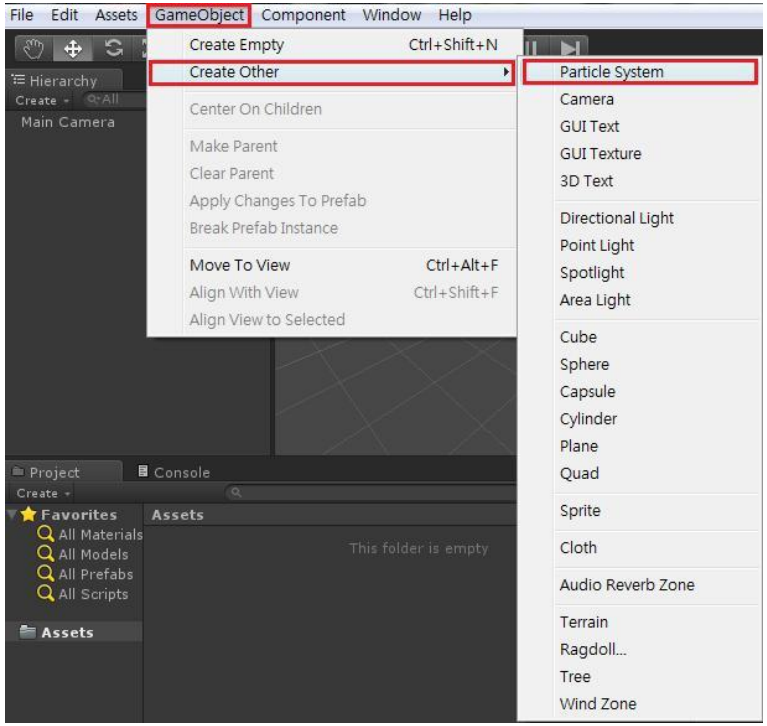
粒子系統

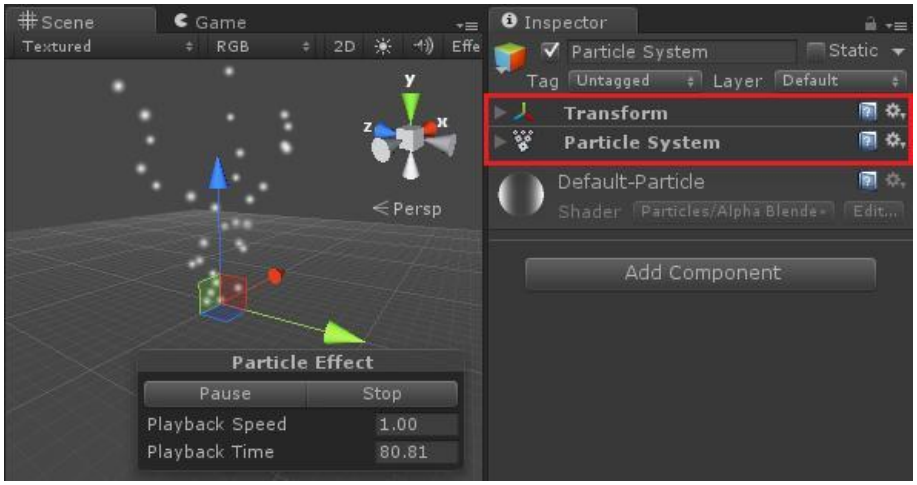






Shuriken粒子系統



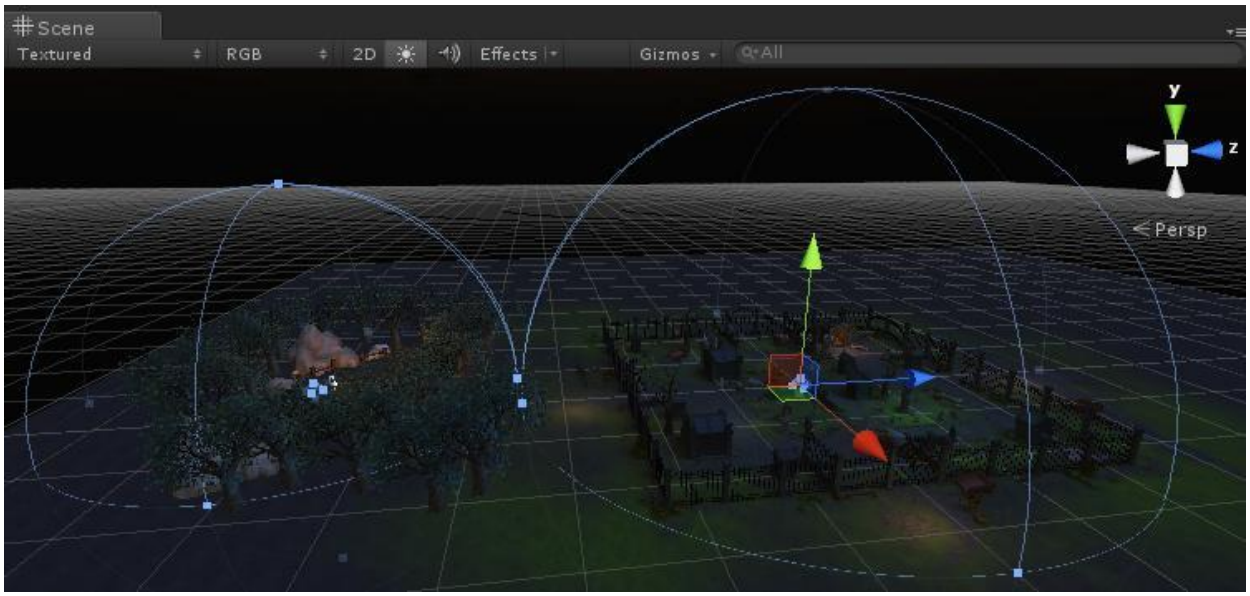


Unity音頻的匯入與使用

- 支援了大多數的音頻格式，包括wav、mp3、aiff、ogg、xm、mod、it與s3m等。

- 二維聲音

- 三維聲音



單元八 靜態場景

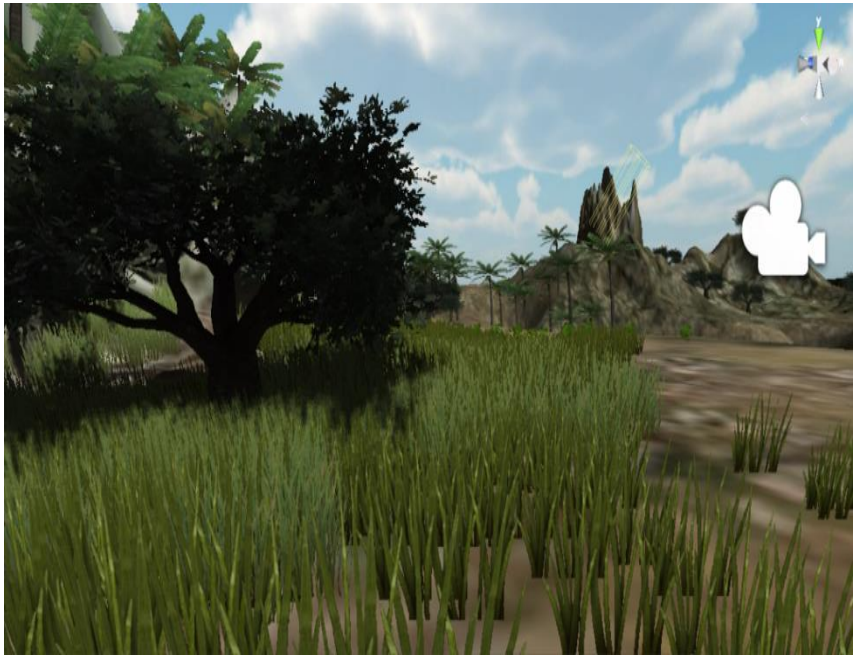
靜態場景光照效果的強化技術

學習重點

- 光照貼圖技術
- 後期屏幕渲染特效

光照貼圖技術

(未烘焙效果)

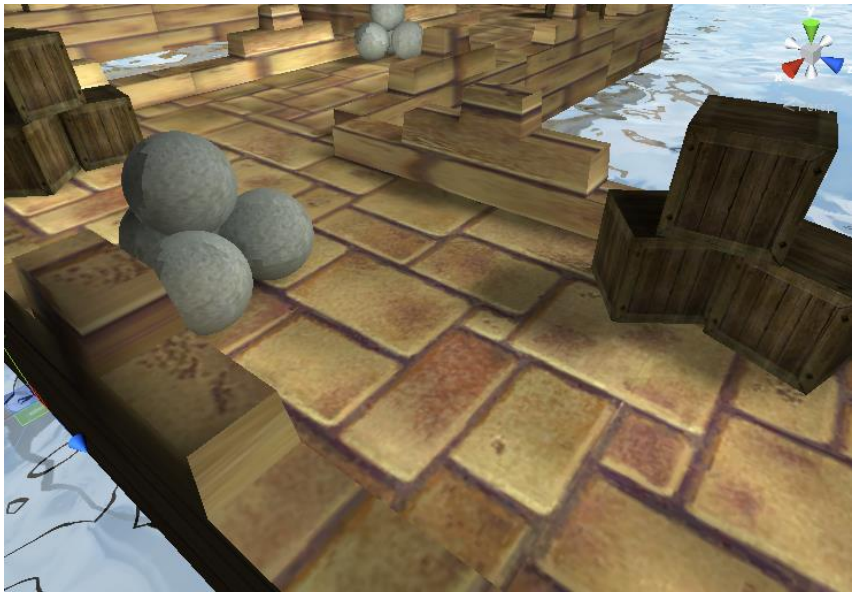


(烘焙效果)

三種烘焙方式

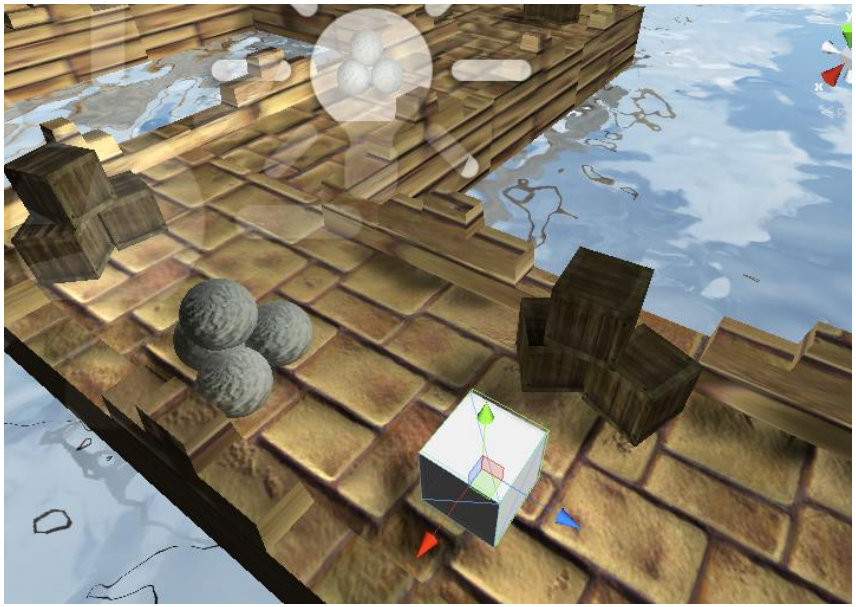
- Single Lightmaps
- Dual Lightmaps
- Directional Lightmaps

Single Lightmaps



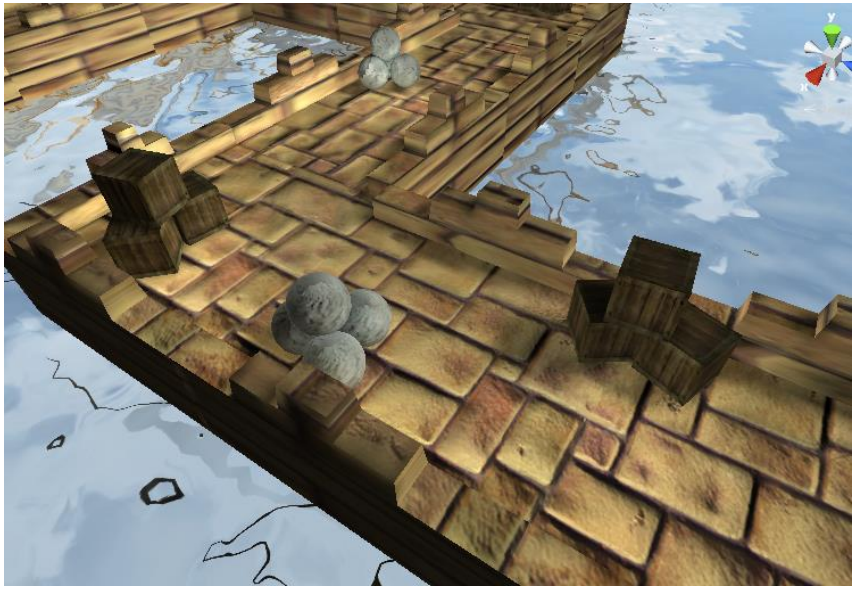
Single Lightmaps是一種簡單的Lightmapping方式，對性能及空間的消耗相對較小，它可以很好地表現靜態場景的的光影效果，但不能夠表現出凹凸貼圖的效果，因為凹凸貼圖要在即時光源的照射下才會產生反應。

Dual Lightmaps



Dual Lightmaps可以在比較大的遊戲場景中表現較多的光影細節，希望多些即時光影，使動態物體和靜態物景的光影融合更為協調，它會將渲染區分為即時和非即時區域，並且烘焙遠近兩種貼圖，離攝影機遠的部分為靜態光照區域，不會表現出太多的細節。

Directional Lightmaps



Directional Lightmaps可以使靜態物體在利用光照貼圖進行光照的同時混合即時的Bump\Spec映射的效果，豐富整個場景的光影細節，讓場景更加地生動逼真，與Dual Lightmaps的區別為Directional Lightmaps是作用於整個場景不受距離的限制，在沒有即時光源下也會產生即時Bump\Spec映射。

後期屏幕渲染特效

Image Effects(圖像特效)主要應用在攝影機上，可以為遊戲畫面帶來豐富的視覺效果，使遊戲畫面更具藝術感和個性。在Unity中，大部分的特效都是混合使用，通過搭配不同的特效就能夠創造出更完美的遊戲畫面效果。

Sun Shafts(陽光射線特效)

Sun Shafts可以用來模擬亮度很高的光源被物體遮擋時所產生的徑向光線散射效果，合理地運用該特效能有效提升遊戲畫面的真實感。

Bloom(泛光特效)

Bloom能夠在增強光暈的同時自動添加高效能的鏡頭眩光。

Tonemapping(色調映射)

Tonemapping可以用來模擬人眼適應環境明暗交替的效果，建議與Bloom搭配使用。

單元九 系統設計

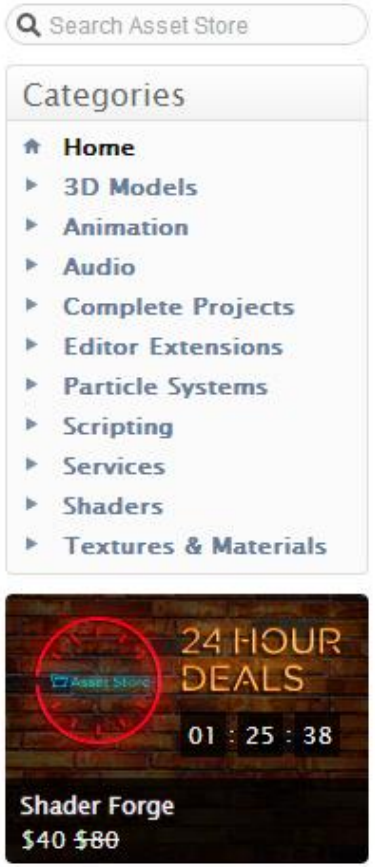
動畫系統應用

學習重點

- 使用Unity的資源商店Asset Store下載資源並匯入模型 。
- 利用第三人稱控制器使人物模型執行動畫並移動。

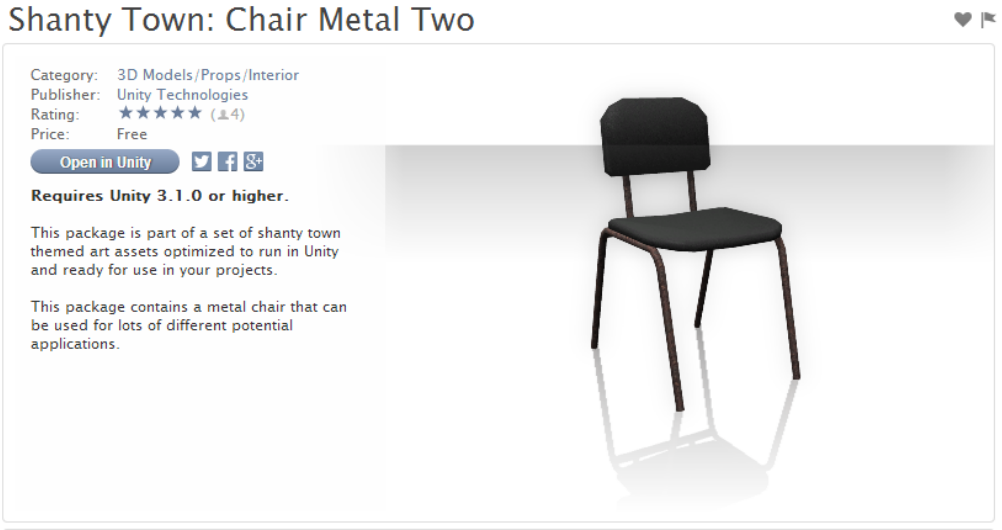
Asset Store資源分類

- 3D Models（3D模型)
- Animation（動畫)
- Audio（音效)
- Complete Projects（完成檔)
- Editor Extensions（編輯器擴充)
- Particle Systems（粒子系統)
- Scripting（腳本)
- Services（服務)
- Shaders (著色器)
- Textures & Materials（紋理與材質)

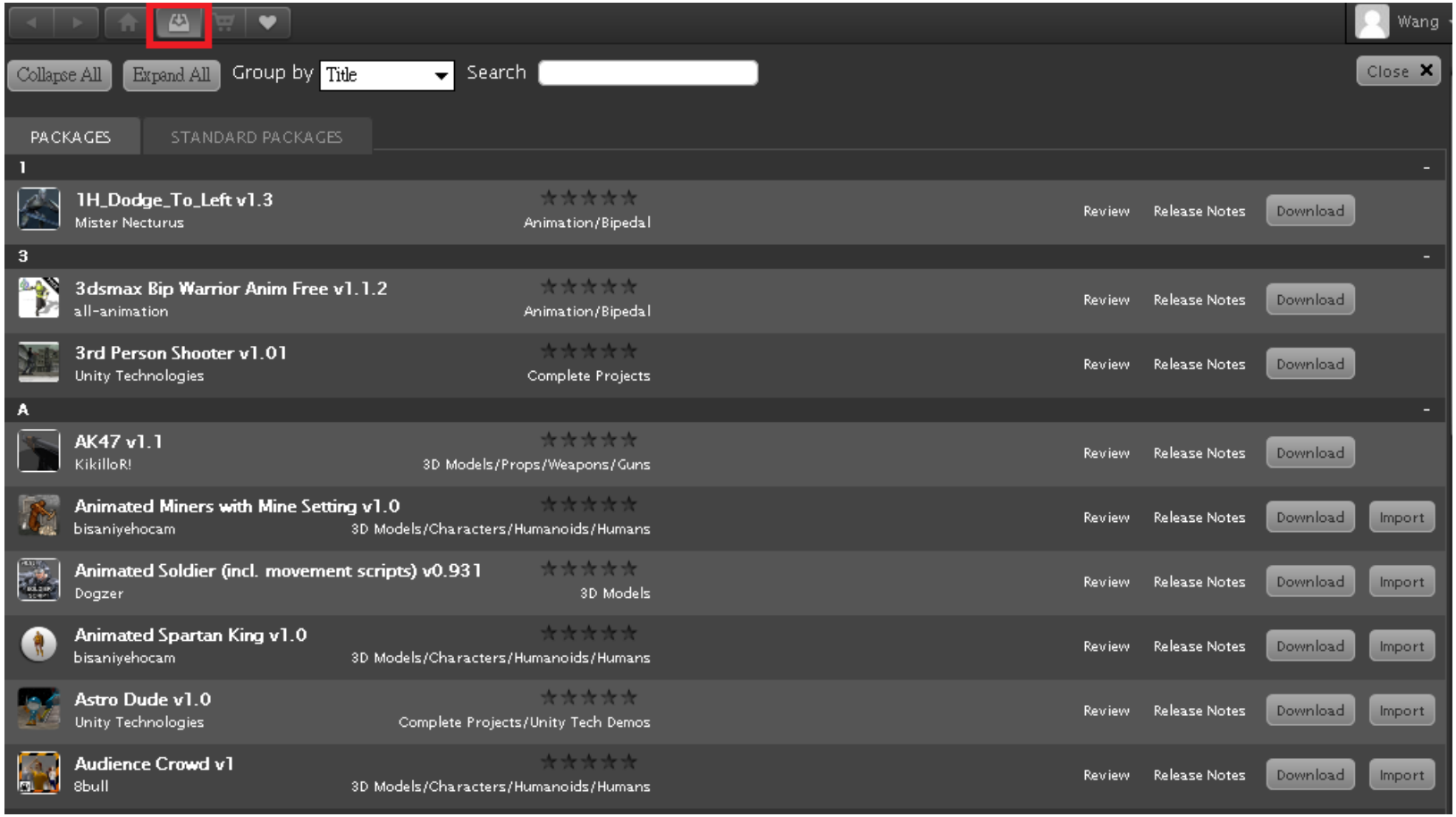


Asset資訊頁面

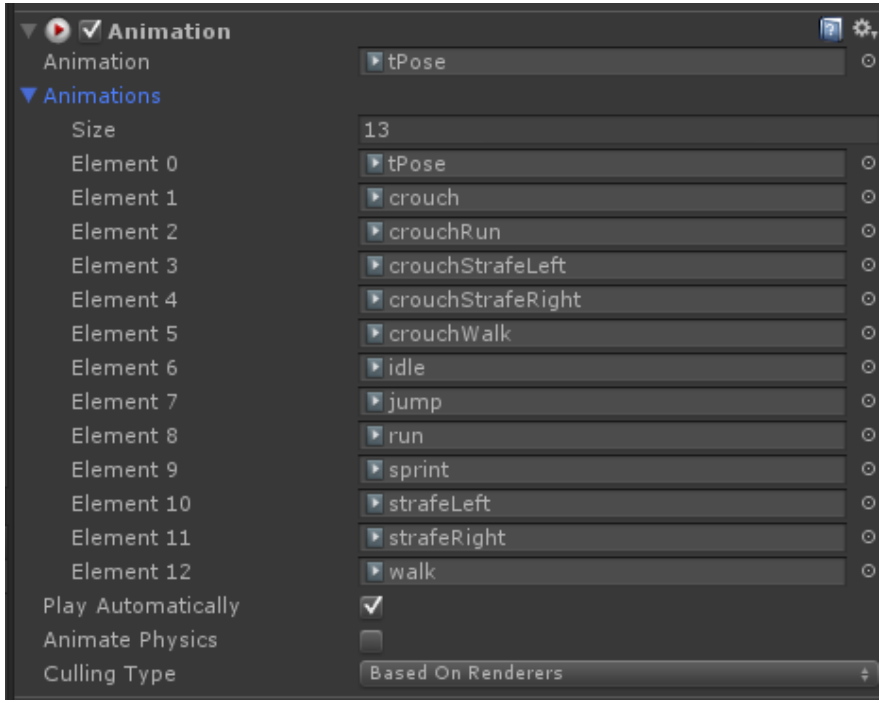
- Category(分類)
- Publisher(出版者)
- Rating(評價)
- Price(價格)
- Version(版本)
- Size(容量)
- Requires(需求版本)



資源歷史紀錄


Wang
Collapse All
Expand All
Group by Title
Search
Close
PACKAGES
STANDARD PACKAGES
1
1H_Dodge_To_Left v1.3
Mister Necturus
Animation/Bipedal
Review
Release Notes
Download
3
3dsmax Bip Warrior Anim Free v1.1.2
all-animation
Animation/Bipedal
Review
Release Notes
Download
3rd Person Shooter v1.01
Unity Technologies
Complete Projects
Review
Release Notes
Download
A
AK47 v1.1
KikilloR!
3D Models/Props/Weapons/Guns
Review
Release Notes
Download
Animated Miners with Mine Setting v1.0
bisaniyehocam
3D Models/Characters/Humanoids/Humans
Review
Release Notes
Download
Import
Animated Soldier (incl. movement scripts) v0.931
Dogzer
3D Models
Review
Release Notes
Download
Import
Animated Spartan King v1.0
bisaniyehocam
3D Models/Characters/Humanoids/Humans
Review
Release Notes
Download
Import
Astro Dude v1.0
Unity Technologies
Complete Projects/Unity Tech Demos
Review
Release Notes
Download
Import
Audience Crowd v1
8bull
3D Models/Characters/Humanoids/Humans
Review
Release Notes
Download
Import

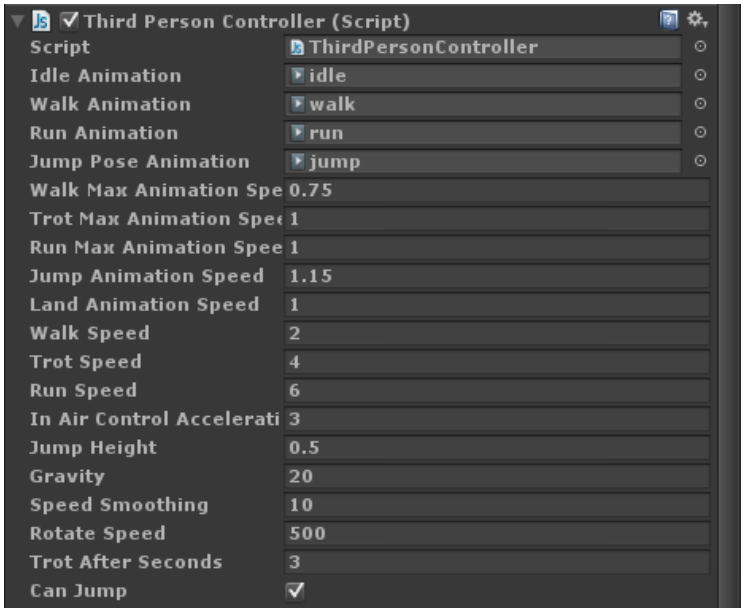
模型本身擁有的動畫資訊



(此模型內擁有13個動畫)

第三人稱控制器可調整變數

- walkMaxAnimationSpeed(走路動畫最大速度)
- trotMaxAnimationSpeed(慢跑動畫最大速度)
- runMaxAnimationSpeed(跑步動畫最大速度)
- jumpAnimationSpeed(跳躍動畫最大速度)
- landAnimationSpeed(著地動畫最大速度)
- walkSpeed(走路速度)
- trotSpeed(慢跑速度)
- runSpeed(跑步速度)
- inAirControlAcceleration(滯空加速度)
- jumpHeight(跳躍高度)
- gravity(重力)
- speedSmoothing(速度平滑度)
- rotateSpeed(旋轉速度)
- trotAfterSeconds(切換跑步速度)。



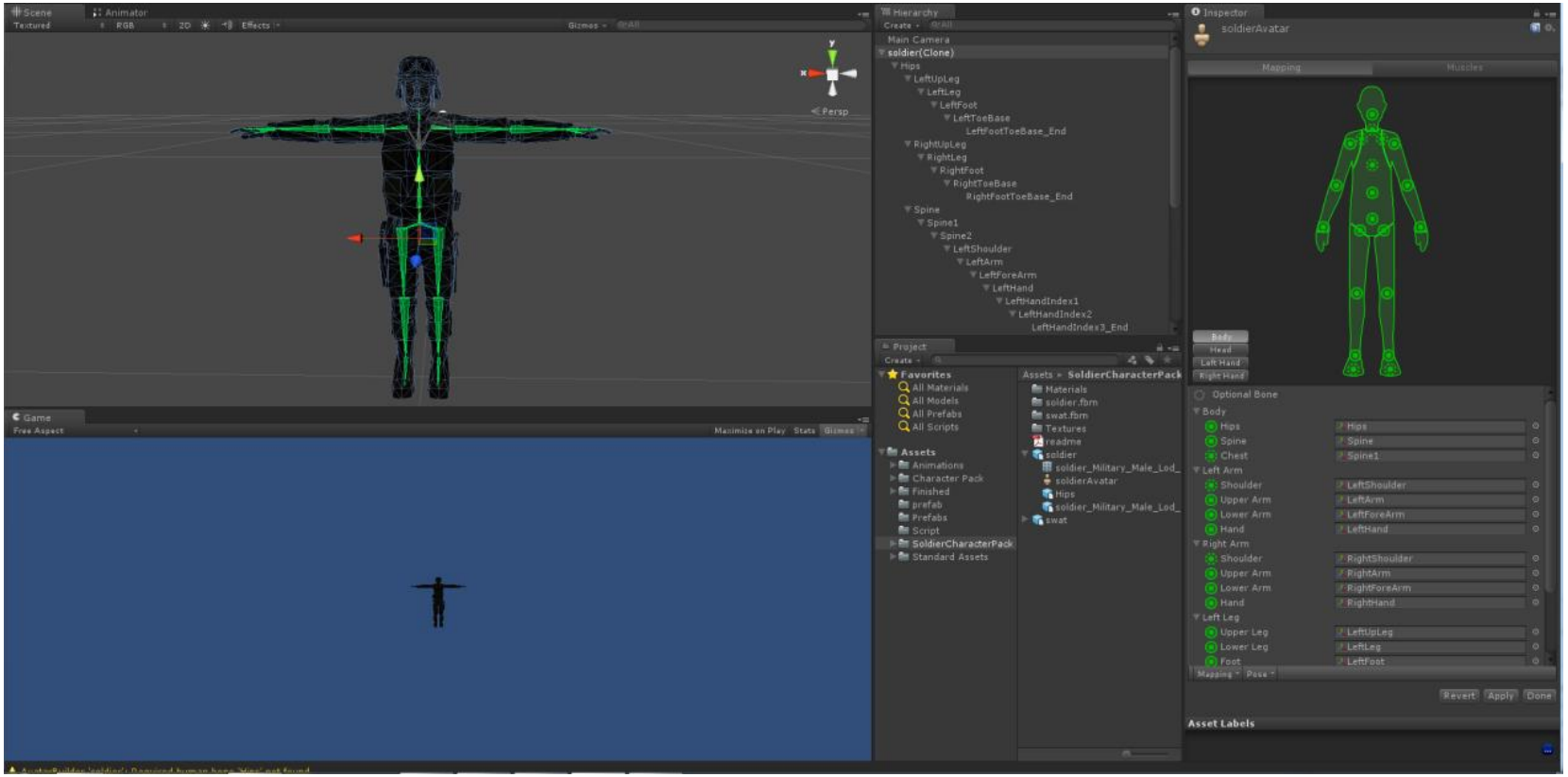
單元十 動畫設計

Mecanim動畫系統

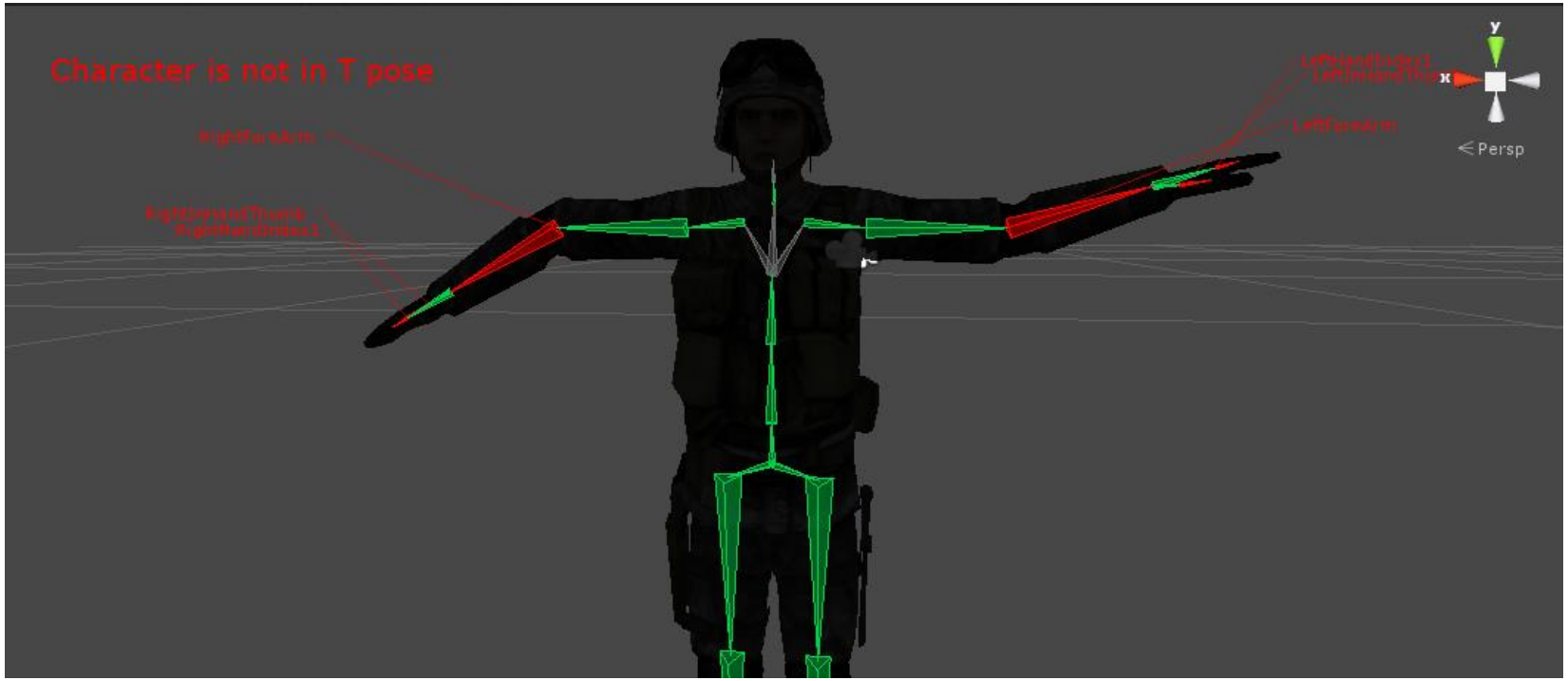
學習重點

- 對人形骨架模型建立Avatar物件
- 建立角色模型的狀態動畫
- 角色模型狀態動畫的切換控制

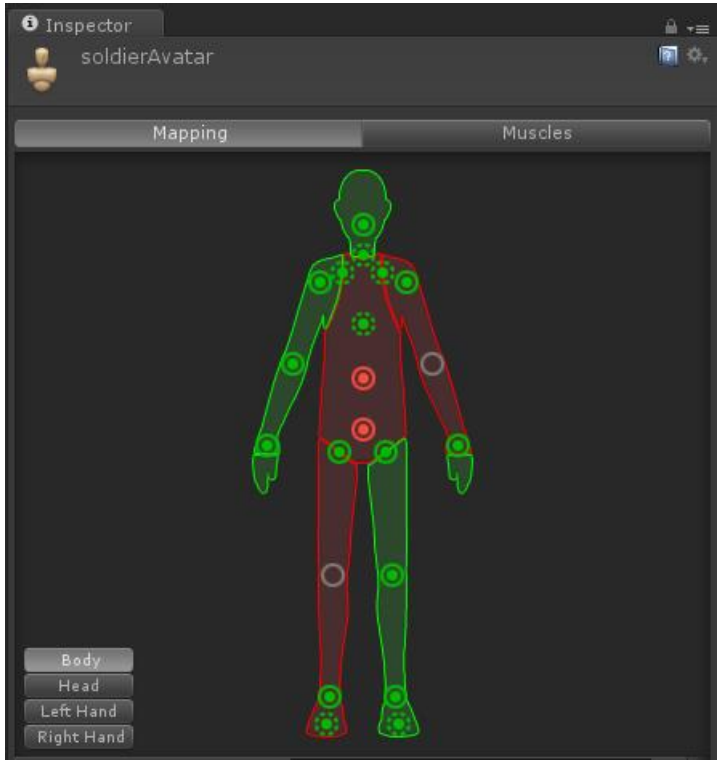
Avatar設置面板

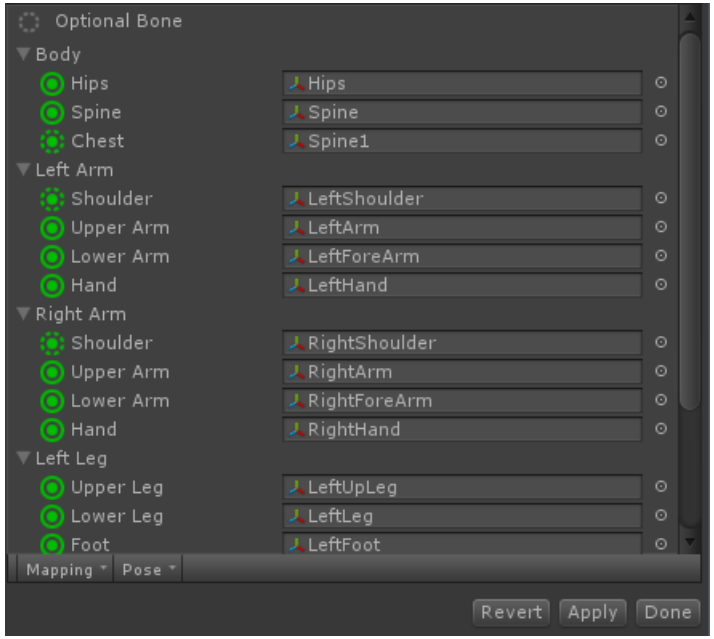

Scene
Animator
Hierarchy
Inspector
soldierAvatar
Mapping
Muscles
Main Camera
soldier(Clone)
Hips
LeftUpLeg
LeftLeg
LeftFoot
LeftToeBase
LeftFootToeBase_End
RightUpLeg
RightLeg
RightFoot
RightToeBase
RightFootToeBase_End
Spine
Spine1
Spine2
LeftShoulder
LeftArm
LeftForeArm
LeftHand
LeftHandIndex1
LeftHandIndex2
LeftHandIndex3_End
Persp
Project
Game
Free Aspect
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Animations
Character Pack
Finished
prefab
Prefabs
Script
SoldierCharacterPack
Standard Assets
Assets ▸ SoldierCharacterPack
Materials
soldier.fbm
swat.fbm
Textures
readme
soldier
soldier_Military_Male_Lod_
soldierAvatar
Hips
soldier_Military_Male_Lod_
swat
Body
Head
Left Hand
Right Hand
Optional Bone
Body
Hips
Spine
Chest
Left Arm
Shoulder
Upper Arm
Lower Arm
Hand
Right Arm
Shoulder
Upper Arm
Lower Arm
Hand
Left Leg
Upper Leg
Lower Leg
Foot
Mapping
Pose
Revert
Apply
Done
Asset Labels

角色關節骨架



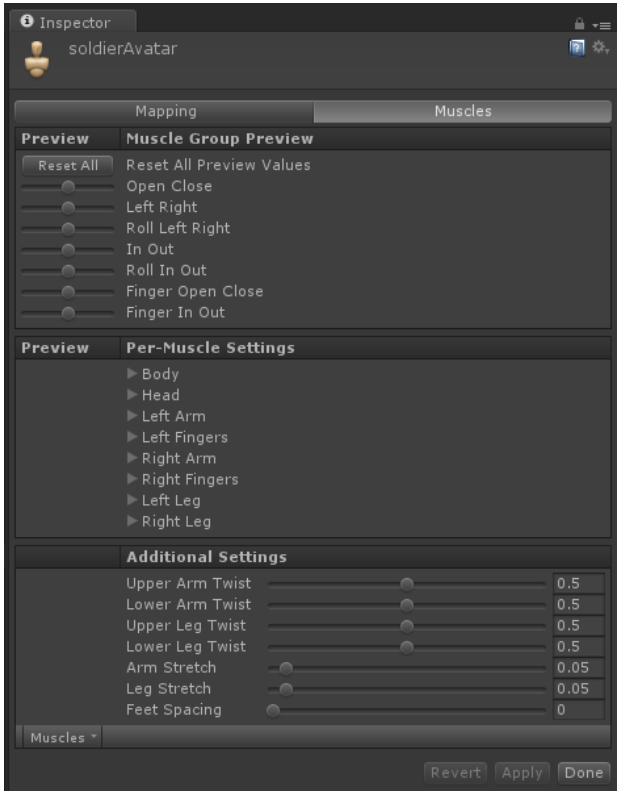
骨架配置面板與骨架名稱對應面板





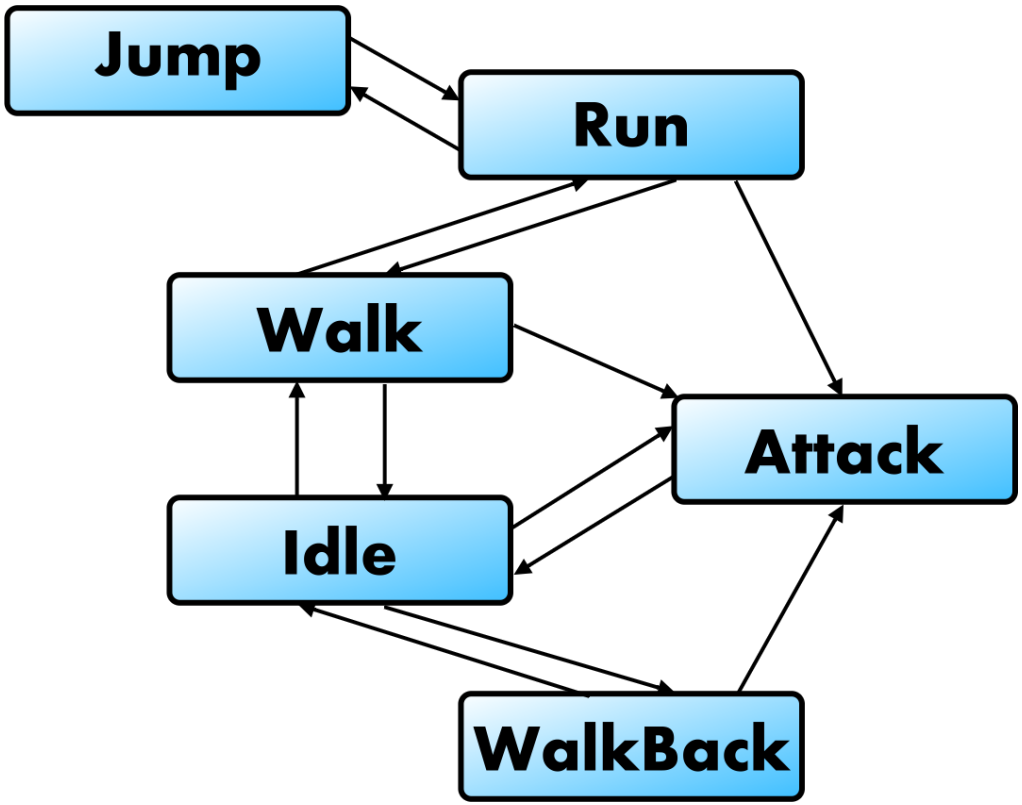
實線圓圈為必須配置的骨架，虛線圓圈為可選擇配置的骨架

Muscles面板

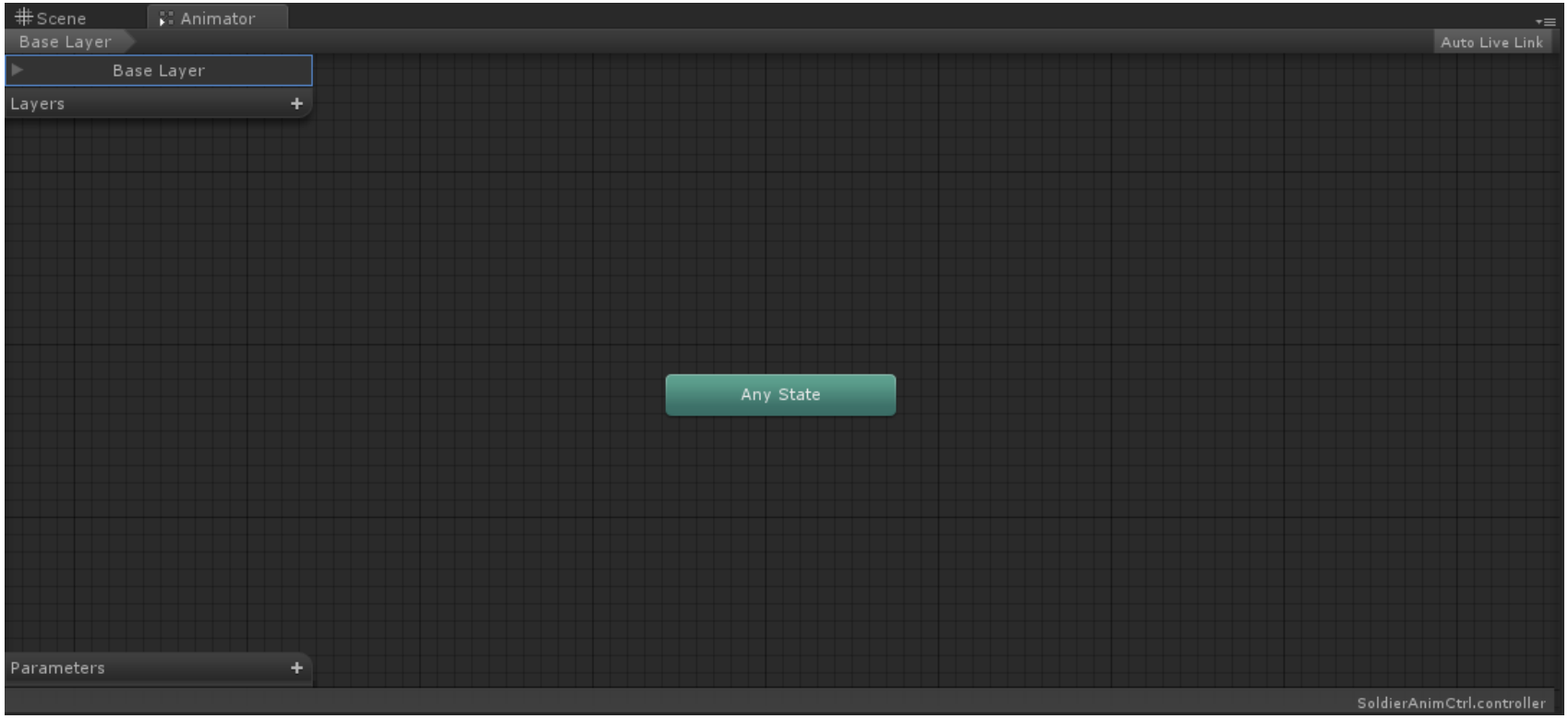


我們可以藉由隨意調整參數看到人物關節的移動及轉動

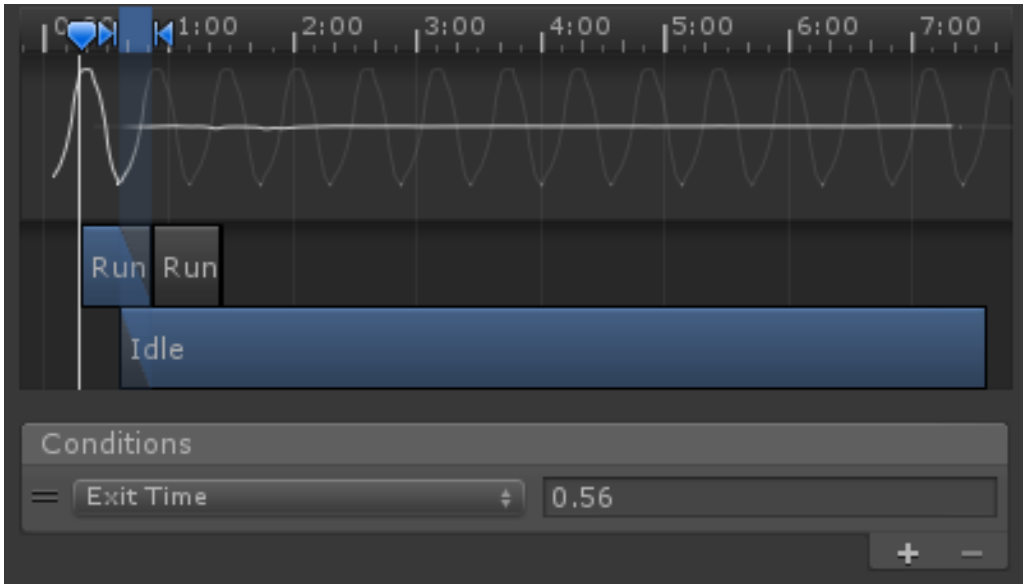
狀態動畫結構示意圖



Animator視窗


Scene
Animator
Base Layer
Auto Live Link
Base Layer
Layers
Any State
Parameters
SoldierAnimCtrl.controller

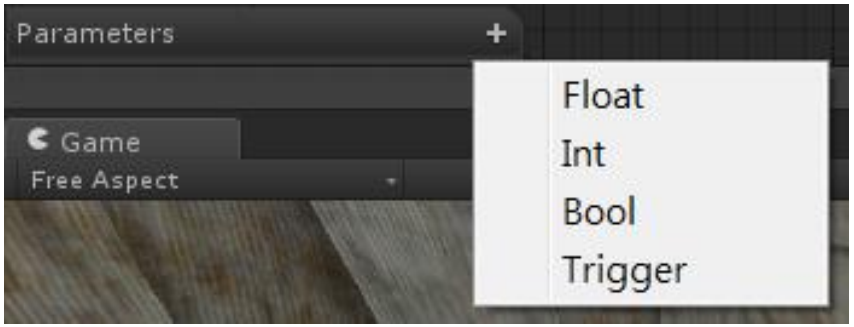
狀態漸變與切換條件



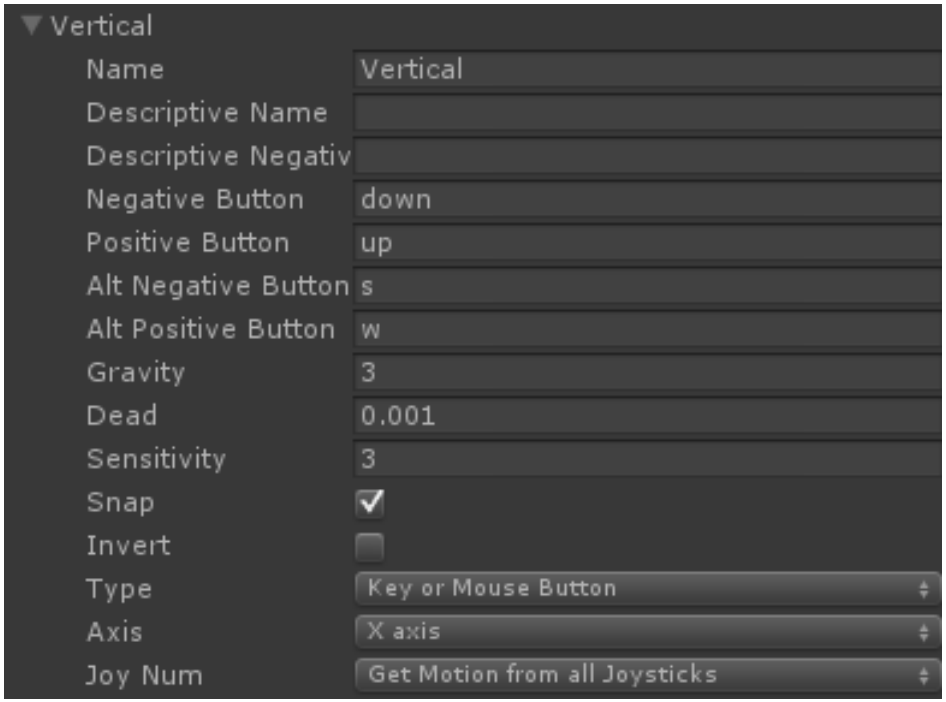
狀態的切換條件可選擇我們所新增的參數，
或是預設條件Exit Time，在撥放一定比例時切換。

Animator中可新增參數

- Float（浮點數)
- Int（整數)
- Bool（布林值)
- Trigger（觸發器)

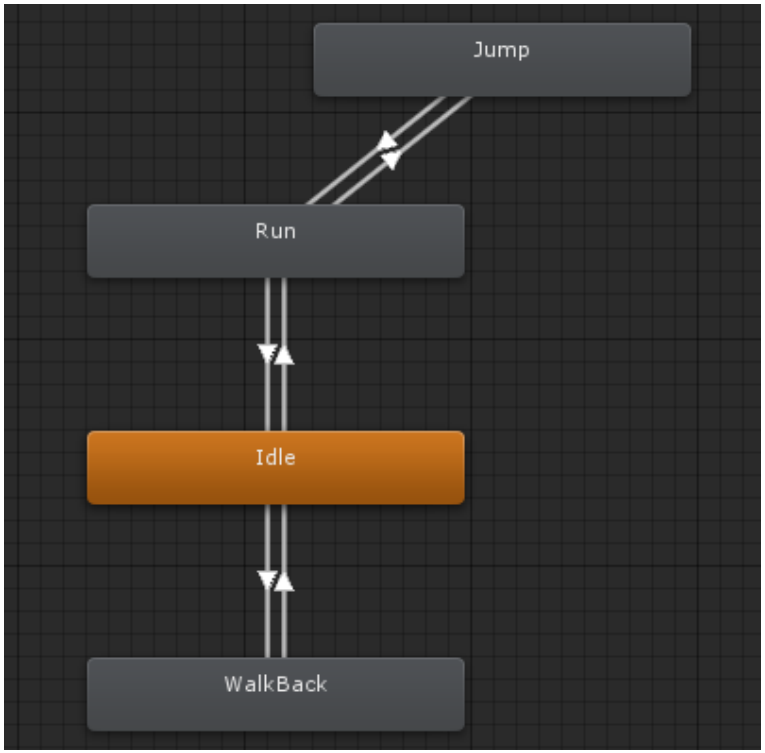


使用Input系統中的參數



(使用Vertical參數時可對應至方向上下鍵)

一個簡單完整的動畫狀態結構



此狀態動畫結構由Idle(待機)、Run(跑步)、WalkBack(退後)、Jump(跳躍)四個狀態動畫所構成。

01-編寫程式介面、流程規劃、時間軸程式介紹

1-1 前言

- Flash在網路中常見的應用：YouTube的影音播放、Facebook上的開心農場與各式小遊戲，還有數不清的網路動畫...等，這些網路上的應用程式都是用Flash開發的。

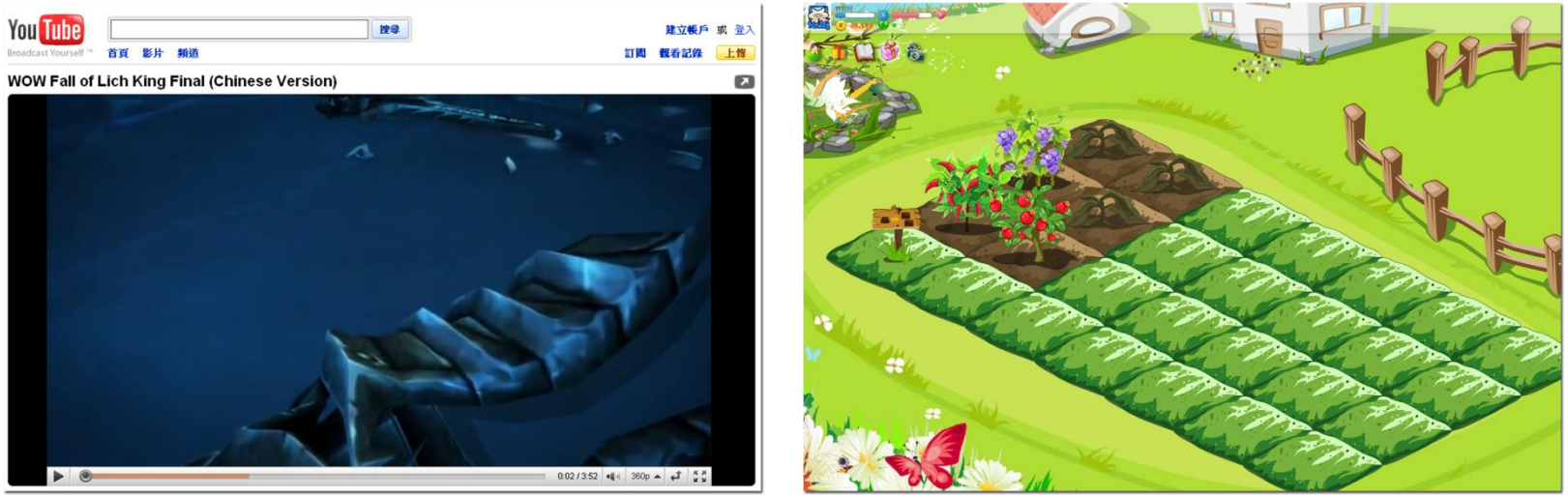


圖1-1 常見的 Flash 應用 (左Youtube 右 開心農場)

圖片出處：左圖http://www.youtube.com 右圖 http://apps.facebook.com/farmgame_tw/index.php

1-2學習目標

- Flash的程式編輯介面。
- 第一支程式：「Hello World!!」。
- 時間軸控制程式的應用。

1-3何謂程式設計

- 程式設計就像我們要使喚屬下做事一樣，我們會用彼此能聽得懂的語言來溝通，並且下達指令，但是如果今天我們的屬下是外國人，那可能還需要一個翻譯來幫忙解釋我們下達的命令，屬下才聽得懂我們要做的事情。

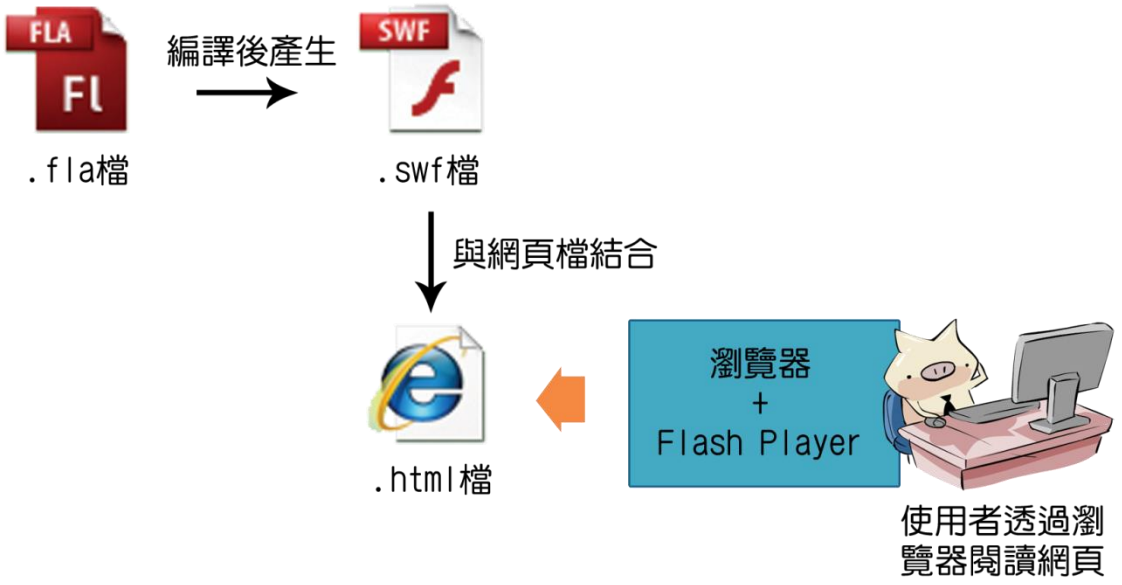
1-3何謂程式設計

使喚屬下做事時，會用彼此能聽得懂的語言來下達指令

電腦只聽得懂機械語言，所以我們必須用程式來溝通

1-4 Flash的程式運作方式

- Flash的編輯程式副檔名是.fla，是我們製作時的原始檔，不能直接放在網路上，網路上看到的應用程式是編譯輸出之後的檔案，副檔名為.swf，透過瀏覽器上的Flash Player播放。

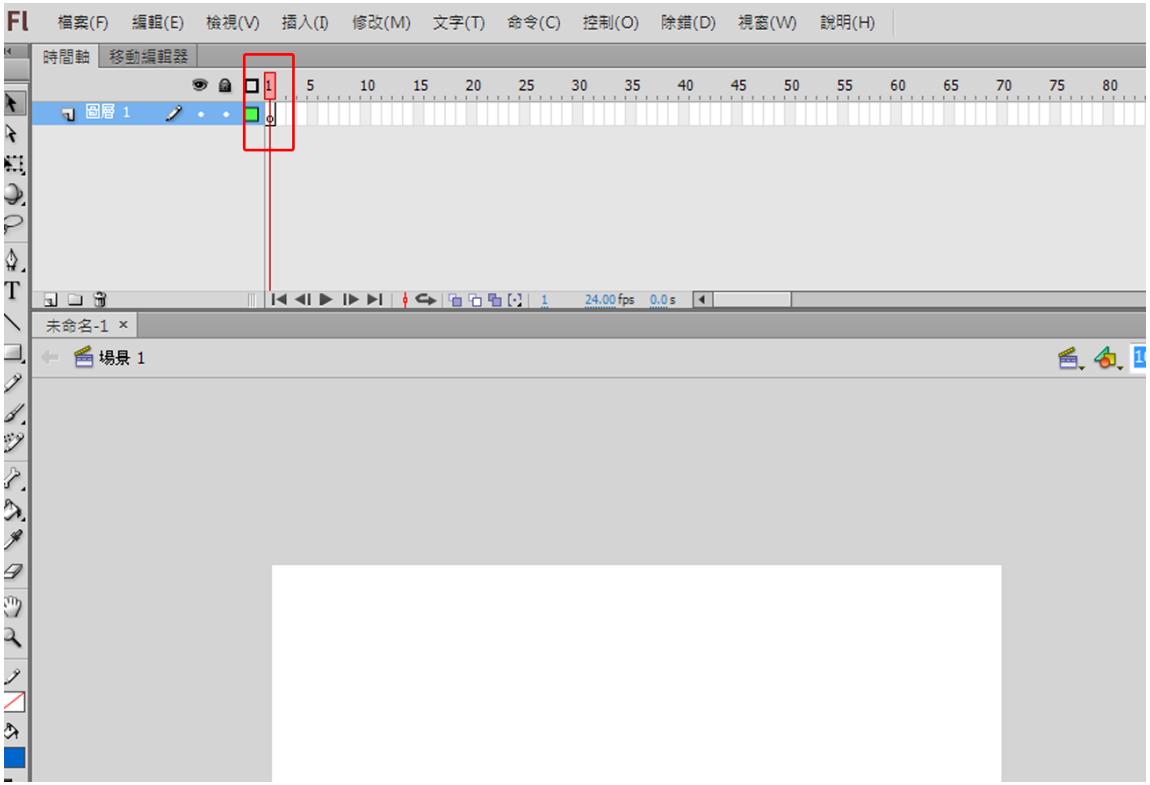


1-5 FLASH介面介紹

- 開啟Flash CS6

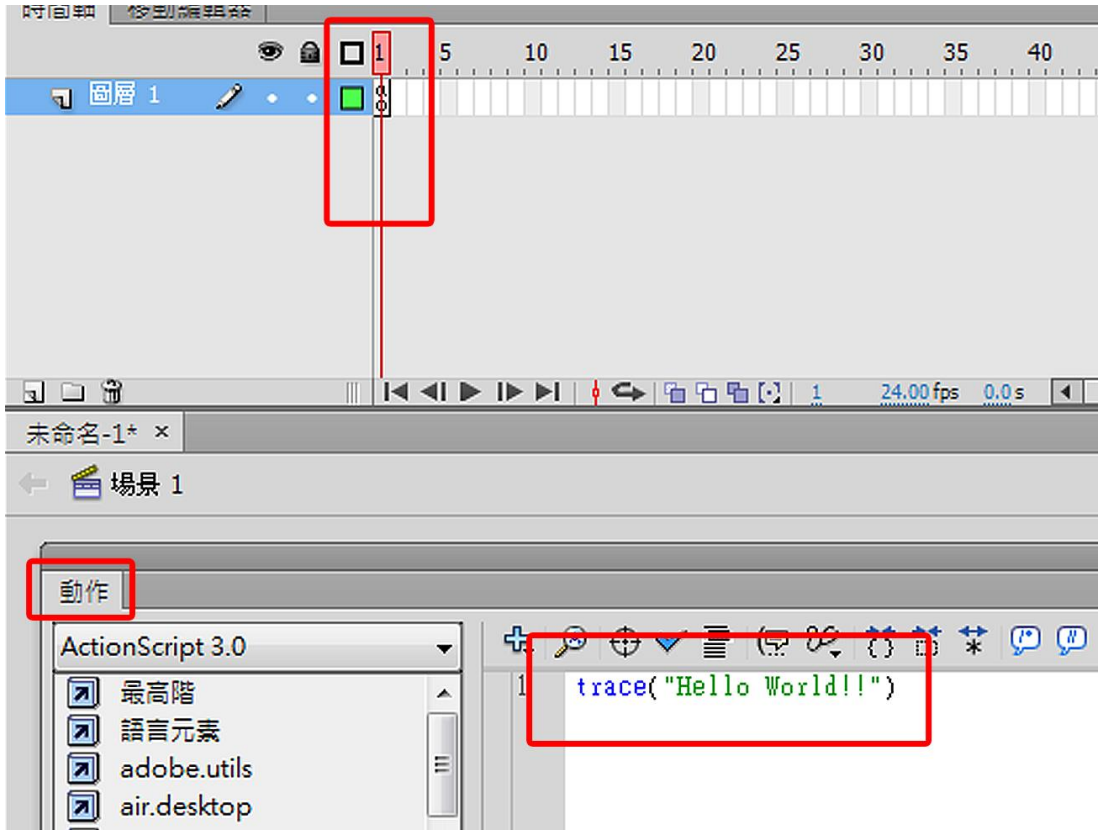
1-5 FLASH介面介紹

- 點選影格



1-5 FLASH介面介紹

- 按下Ｆ９呼叫出動作面版



1-5 FLASH介面介紹

- 寫程式常用的的幾個功能：

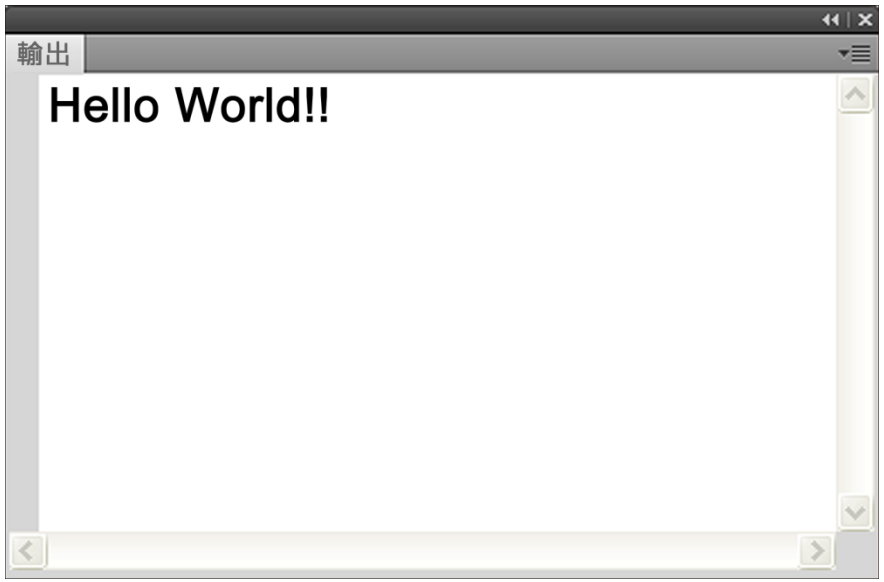
「尋找」：搜尋程式碼裡的文字。
「檢查語法」：檢查語法是不是有錯誤。
「自動格式化」：自動整理程式碼，可以幫忙把程式碼排整齊。
「顯示隱藏工具箱」：開啟或關閉左邊的函式工具箱(關起來可以有比較大的編寫空間)

1-6 第一個AS程式

- ActionScript中最基本的輸出指令「trace」。
- 「trace」指令有兩種使用方式：
- 用法一：trace("輸出字串文字"); → 輸出我們想要輸出的字串。
- 用法二：trace(變數); → 把變數的數值輸出。

1-6 第一個AS程式

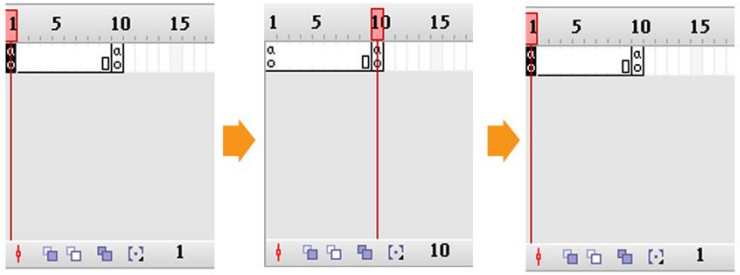
- 開啟「動作」面版：
- 輸入trace("Hello World!!");
- 先壓著Ctrl鍵，再按下Enter鍵，得到以下結果。



※要注意的一點是「trace」這個指令要輸出的內容只有在測試的時候看得到。

1-7 特殊的時間軸概念

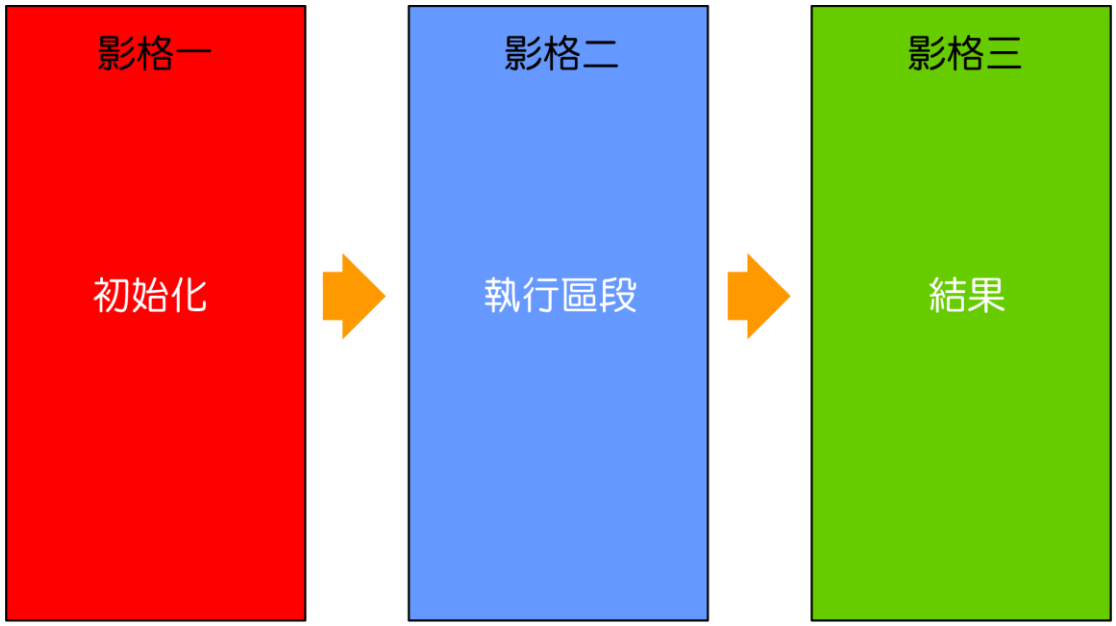
- Flash的AS程式的幾個重要觀念：
- 要注意程式是否需要停止在影格上，如果未下指令停止，將會順播。



←時間軸會自動重覆播放

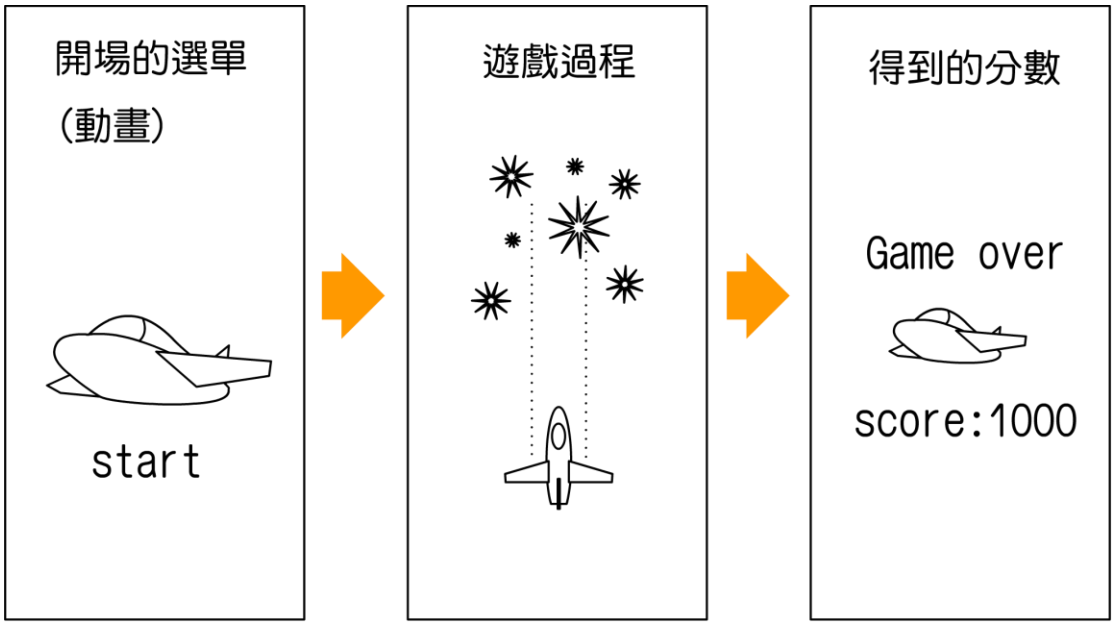
- 程式寫在越左邊的影格，會越優先執行。
- 每個關鍵影格上的元件是獨立的。

1-8 流程規劃



- 第一格是程式的初始化包含所有變數與函式。
- 第二格是程式的執行區段做各種程式的運算。
- 第三格是程式執行的結果。

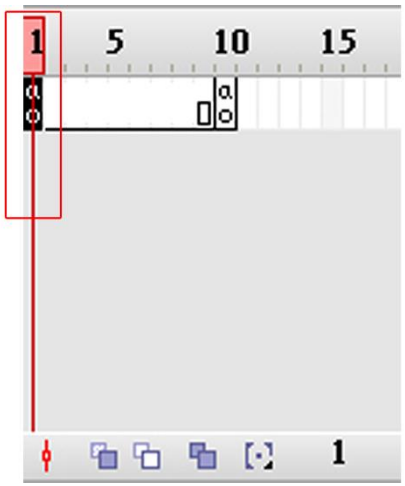
1-8 流程規劃



- 射擊遊戲的三個流程

1-9 時間軸控制程式

- 所謂的時間軸控制程式就是用來控制播放磁頭的指令，我們可以藉由這些指令，讓播放磁頭停止或是跳到我們指定的影格，舉個最簡單的例子，這些控制程式其實就像錄放影機上的按鈕，有播放、有暫停...等。



1-10 stop與gotoAndSop / gotoAndPlay指令

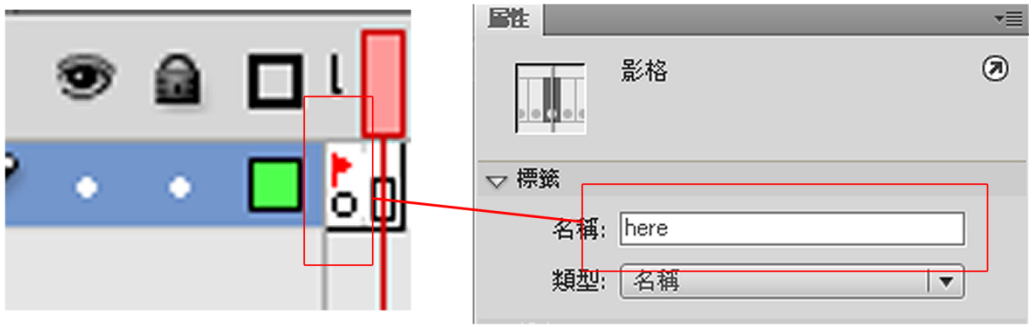
- stop(); 指令後面的括弧不可少。
- 當我們在影格上下了「stop」的指令之後，播放磁頭一旦讀取到，就會停在這個影格上。
- 這個指令只會停止時間軸的播放，並不會停止程式編譯的執行。

1-10 stop與gotoAndSop / gotoAndPlay指令

- gotoAndStop(參數);　gotoAndPlay(參數);
- 「gotoAndSop / gotoAndPlay」這兩個指令是用來讓播放磁頭可以跳到我們希望它讀取的影格，差別在於「gotoAndStop」是讓播放磁頭到了那個影格之後停止動作，不繼續播放讀取右邊的影格，而「gotoAndPlay」則是到了那個影格之後，還會繼續播放右邊的影格。

1-10 stop與gotoAndSop / gotoAndPlay指令

- gotoAndStop(參數);　gotoAndPlay(參數);
- 如果輸入的是數字就會將播放磁頭移動到數字指定的影格。
- 如果輸入的參數是字串就會將播放磁頭移動到與字串相符的影格標籤名稱的影格。



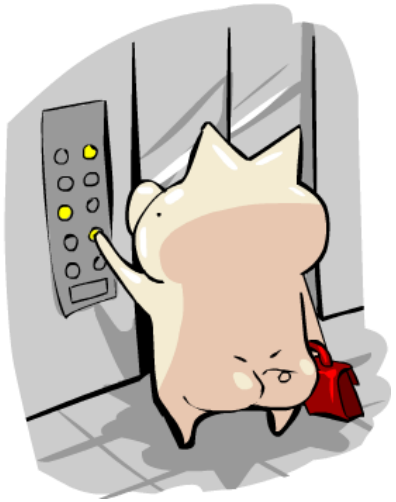
1-11習題

1. 解釋fla檔與swf檔的差別?
2. 舉出一項目前flash的應用範例。
3. trace(a)與trace(“a”)有何差別?
4. stop指令是否需要輸入參數?
5. 如何發佈測試檔案(輸出swf檔)?

02-FLASH的事件(event)-按鈕程式介紹

2-1前言

- Flash程式與使用者有著非常密切的互動，而連結使用者與Flash的橋樑，就是按鈕(button)，所有Flash網頁中可以點選的動作，都是由按鈕觸發的。
- 就像我們搭電梯時，可以點選按鈕，讓電梯帶我們到想要去的樓層。

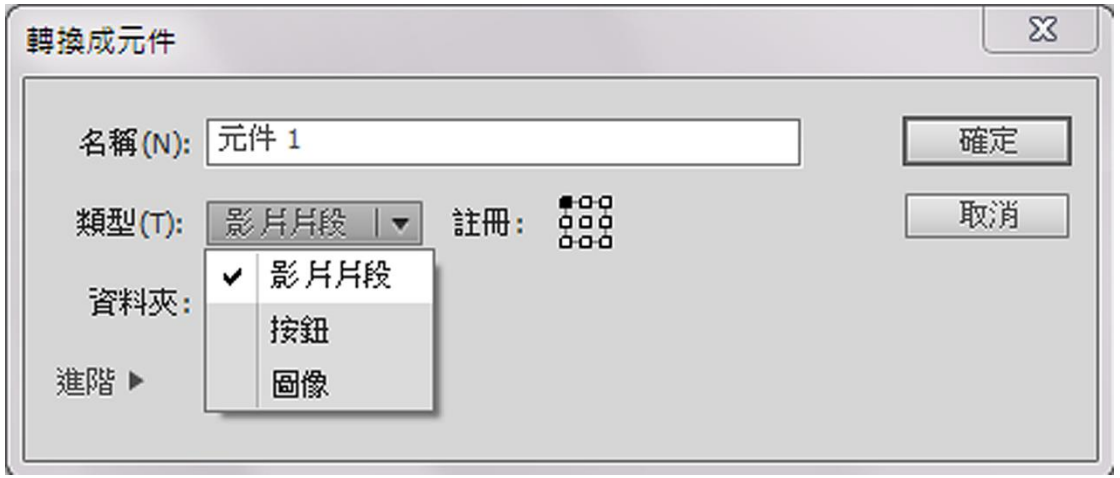


2-2學習目標

- 了解影片片片段與按鈕的差別。
- 編寫按鈕的程式語法。
- 按鈕中的事件。
- Flash中的超連結。

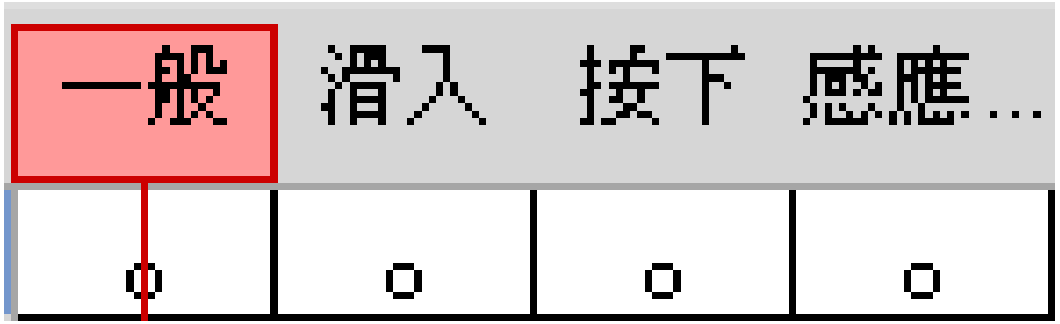
2-3簡介影片片段（MovieClip）與按鈕（button）的差別

- 建立元件的時候，我們可以發現元件的選項有三個，一個是影片片段（MovieClip），一個是按鈕（button）還有一個是圖像(graphic)。



2-3簡介影片片段（MovieClip）與按鈕（button）的差別

- 雙點元件進入之後可以看出來影片片段裡具備時間軸，但是按鈕卻只有四個大大的影格，分別寫著一般、滑入、按下、感應區。

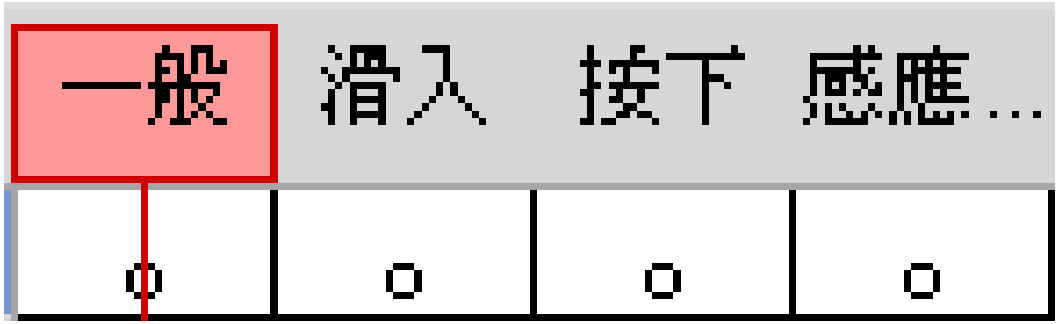


2-3簡介影片片段（MovieClip）與按鈕（button）的差別

- 藉由使用者的觸發（滑入或是點擊、按壓等），我們可以秀出按鈕的各種不同的形狀。

2-3簡介影片片段（MovieClip）與按鈕（button）的差別

- 雙點元件進入之後可以看出來影片片段裡具備時間軸，但是按鈕卻只有四個大大的影格，分別寫著一般、滑入、按下、感應區。



2-4按鈕程式的寫法

◎案例**2-1**：

程式碼：
1. function btnf(event:MouseEvent){ 2. //按鈕被點選時要執行的動作 3. trace("Hello!"); 4. } 5. btn.addEventListener(MouseEvent.CLICK,btnf);

行數	程式碼的功用
1	建立按鈕「btn」被點選時要執行的動作函式（函式的觀念可以參閱第七章）。
2	註解文字。
3	輸出字串「Hello!」。
5	將按鈕「btn」加入事件監聽，只要使用者點選就觸發動作，而監聽的事件是滑鼠點選事件，要執行的動作函式是「btnf」。

2-4按鈕程式的寫法

- 在AS3中，替按鈕加入功能的步驟：

1. 宣告一個**函式(function)**，它的輸入參數一定要是event:MouseEvent（關於函式的詳細說明，我將留在第七章解說）。
2. 將按鈕要使用的功能寫在**函式**裡。
3. 將按鈕加入**事件監聽（addEventListener）**，並且指定在什麼情況下（MouseEvent.CLICK）才會觸發，觸發時的函式名稱為何（範例2-1中的函式名稱為「btnf」，這是我們自己命名的函式）。

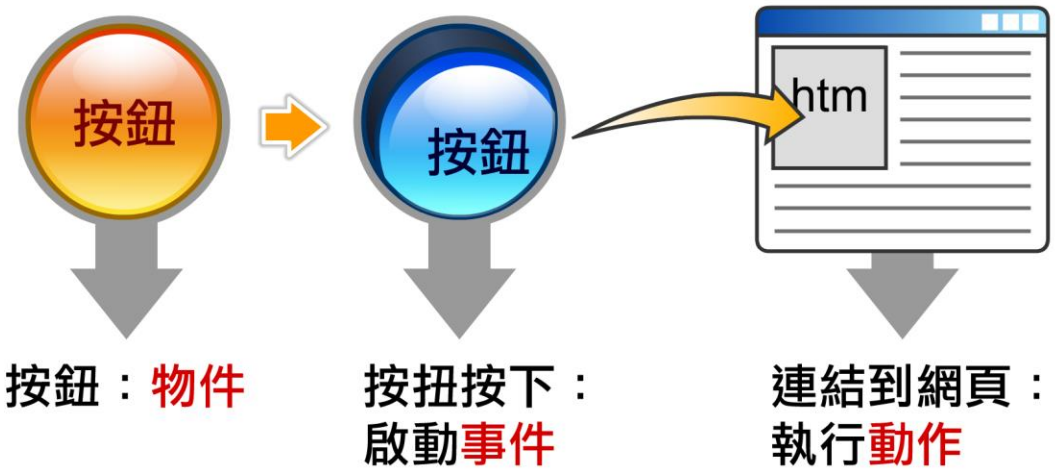
2-5 函式（function）

- 可以把函式想像成一個獨立工作的小程式，它與外部的聯繫只有依靠我們傳進去的參數，而在按鈕程式中，我們輸入的參數就是觸發狀態，也就是滑鼠事件，像點選，滑過...等，都是滑鼠事件，而函式接受到命令之後就會開始執行「{}」內程式碼所要求的動作。

```
function 函式名稱(參數){
    要執行的動作
}
```

2-6事件（event）

- 舉例說明：比方網頁上有個按鈕它被按下之後會連結到另一個網頁，那麼被「滑鼠按下」這個狀態就是「事件」，而按鈕就是「物件」，而執行的「動作」就是連結到另一個網頁。



2-7按鈕常見的幾種事件

事件	代表的意義
MouseEvent.CLICK	滑鼠左鍵點擊一下之後放開。
MouseEvent.DOUBLE_CLICK	滑鼠左鍵快速點擊兩下之後放開。
MouseEvent.MOUSE_DOWN	滑鼠左鍵按下沒有放開。
MouseEvent.MOUSE_MOVE	滑鼠指標在畫面上移動，並且沒有按壓任何按鍵。
MouseEvent.MOUSE_OUT	滑鼠指標從按鈕的感應區上移開。
MouseEvent.MOUSE_OVER	滑鼠指標停在按鈕的感應區上。
MouseEvent.MOUSE_UP	滑鼠左鍵放開（通常跟MOUSE_DOWN狀態搭配）。
MouseEvent.MOUSE_WHEEL	滑鼠滾輪的動作。
MouseEvent.ROLL_OUT	滑鼠左鍵按下的情況下，移動滑鼠指標離開按鈕感應區。
MouseEvent.ROLL_OVER	滑鼠左鍵按下的情況下，移動滑鼠指標到按鈕感應區中。

2-8其它常見的事件

事件	代表的意義
Event.ENTER_FRAME	根據影格速率的設定（預設值是1/12秒），每經過這段時間，就會引發這個事件。
Event.ADDED	將物件加入時就會引發這個事件（比方從元件庫呼叫元件進場景的時候）。
Event.REMOVED	移除物件時就會引發這個事件（比方從場景上移除元件的時候）。
Event.CHANGE	當文字欄位內的資料被改變時，會引發這個事件。
Event.COMPLETE	當外部資料載入完畢的時候引發這個事件。
Event.SOUND_COMPLETE	當聲音播放完畢的時候引發這個事件。
Event.RESIZE	當視窗被縮放導致動畫大小改變時會引發這個事件。
Event.FULLSCREEN	當開啟或離開全螢幕模式的時候會引發這個事件。
Event.MOUSE_LEAVE	使用者的滑鼠指標移動到Flash外面時會引發這個事件。

2-9加入事件監聽（addEventListener）

- 當我們在寫函式的時候就已經把按鈕要執行的功能都寫好了，但是電腦並不知道什麼才要執行這些功能，因此事件監聽會一直不停地觀察場景上物件狀態的變化只要加入監聽的物件出現指定的事件（例如：按鈕被按下），就會執行命令（例如：連結到另一個網頁）。
- 事件監聽的語法如下：

加入監聽的物件（按鈕或影片片段）.addEventListener(事件,函式名稱);

2-10移除事件監聽（removeEventListener）

- 解除事件監聽，就必需使用移除事件監聽的指令removeEventListener，基本上這個指令就是跟加入事件監聽addEventListener是配成一對的，如果我們要解除某項物件的監聽，只要把addEventListener指令改成removeEventListener就可以了。

移除監聽的物件（按鈕或影片片段）.removeEventListener(事件,函式名稱);

2-11超連結

- AS3之中取消了getURL這個函式，改成以下的程式碼來進行超連結

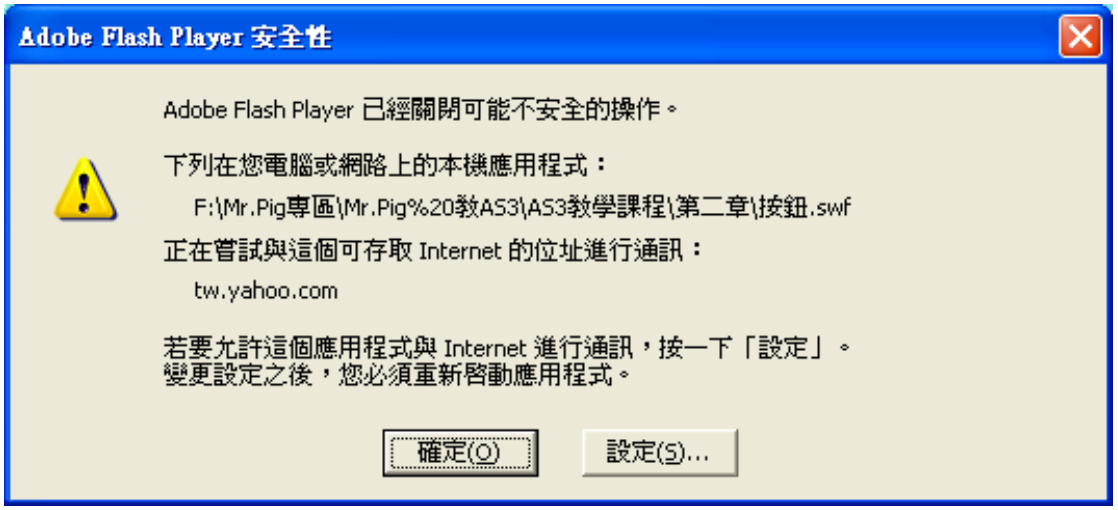
◎案例2-4：

程式碼：
1. var linkURL:String="http://tw.yahoo.com"; 2. var linkRQ:URLRequest=new URLRequest(linkURL); 3. navigateToURL(linkRQ);

行數	程式碼的功用
1	首先我們要先宣告一個變數linkURL，將要連結的網址先存到變數之中。
2	建立一URLRequest物件，並且將linkURL加入
3	使用navigateToURL函式開啟連結。

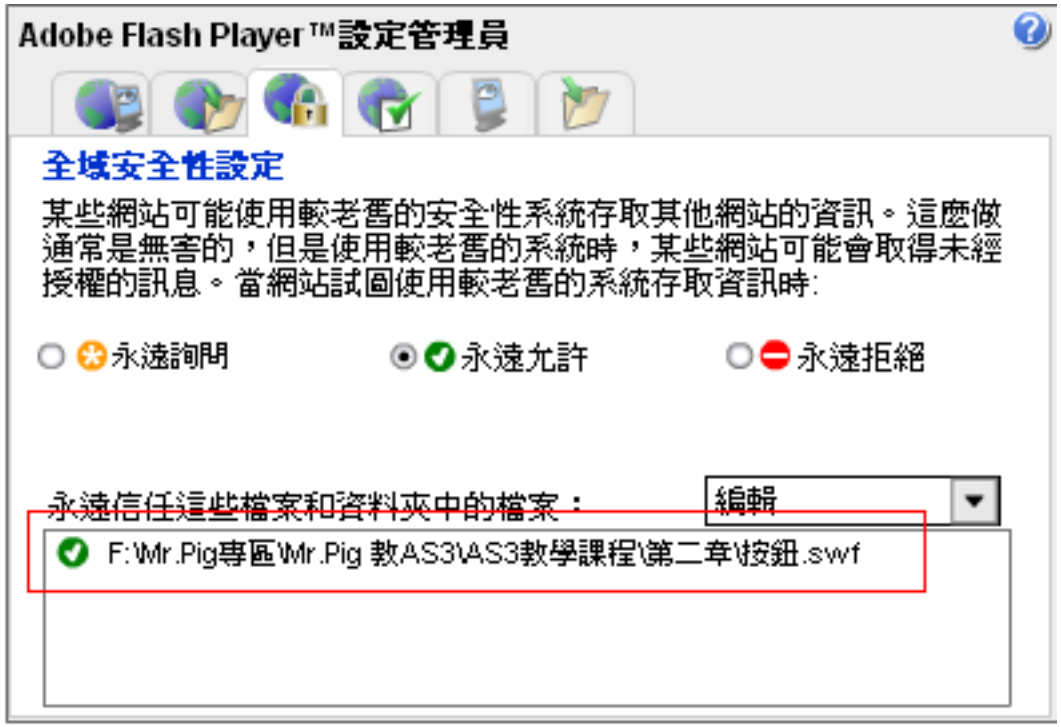
2-12本機測試超連結的問題

- 當我們製作完超連結的按鈕，直接開啟發佈完成的swf檔時，我們會發點選按鈕卻不能作外部連結，而且會跑出安全性的警告。



2-12本機測試超連結的問題

- 按下設定之後會到下列的畫面，再將swf檔的位置新增到下面的方框（如下圖）。



2-13章節案例：連結網站的按鈕

◎目標：

可以連結到各大網站的按鈕。

◎程式原理：

利用事件與事件監聽，建立可以讓使用者點選之後開啟連結的按鈕。

2-13章節案例：連結網站的按鈕

程式碼

```
1.  //超連結到Yahoo的按鈕
2.  function link_yahoo(event:MouseEvent) {
3.    var linkURL:String="http://tw.yahoo.com";
4.    var linkRQ:URLRequest=new URLRequest(linkURL);
5.    navigateToURL(linkRQ);
6.  }
7.  btn_yahoo.addEventListener(MouseEvent.CLICK,link_yahoo);
```

行數	程式碼的功用
1	註解文字。
2~6	函式link_yahoo，告訴按鈕btn_yahoo當他被點選時，要連結到外部的網站。
7	讓按鈕btn_yahoo加入事件監聽，當它被點選時，就會執行函式link_yahoo中的動作。

2-13章節案例：連結網站的按鈕

- 完成檔

2-14 習題

1. （　）一個按鈕元件有幾種狀態？
（Ａ）1種（Ｂ）2種（Ｃ）3種（Ｄ）4種。
2. （　）下列哪一種元件類型無法用程式控制？
（Ａ）影片片段（Ｂ）按鈕（Ｃ）圖像（Ｄ）以上皆可。
3. （　）當按鈕被滑鼠滑過時，會觸發下列哪種事件？
（Ａ）MouseEvent.DOUBLE_CLICK（Ｂ）MouseEvent.MOUSE_MOVE（Ｃ）MouseEvent.MOUSE_OVER（Ｄ）MouseEvent.MOUSE_UP。
4. （　）下列哪一個指令可以移除事件監聽？
（Ａ）removeEventListener（Ｂ）addEventListener（Ｃ）navigateToURL（Ｄ）URLRequest。
5. （　）使用一個視窗參數，可以讓「**navigateToURL**」在新的視窗開啟連結？
（Ａ）_self（Ｂ）_blank（Ｃ）_parent（Ｄ）_top。

03-movieclip屬性、整體架構，路徑

3-1前言

- 動畫的原理利用的是人類眼睛的一種特性，叫作「視覺暫留」。
- 向量動畫最特別的地方就是繪製完物件後，再將物件建立為元件，利用電腦去計算動態的變化。
- 建立為元件(MovieClip)的圖形在Flash中被視為是一種物件導向」的概念。

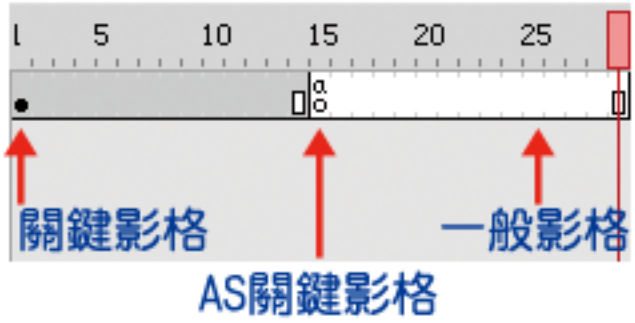
AS3重新以物件來打造

3-2學習目標

- 認識Flash的整體架構。
- 了解何謂物件導向程式。
- 認識影片片段（MovieClip）與「Sprite」。

3-3場景與影片片段

- 開啟一個新檔案的時候，中間有一個編輯區塊，這個區塊被稱之為場景（stage）。
- 影格可以分為一般影格與關鍵影格，關鍵影格影格上會有一個○，而所有的程式都只能寫在關鍵影格上，要注意用程式呼叫元件時，該元件一定要在這個關鍵影格上。



3-3場景與影片片段

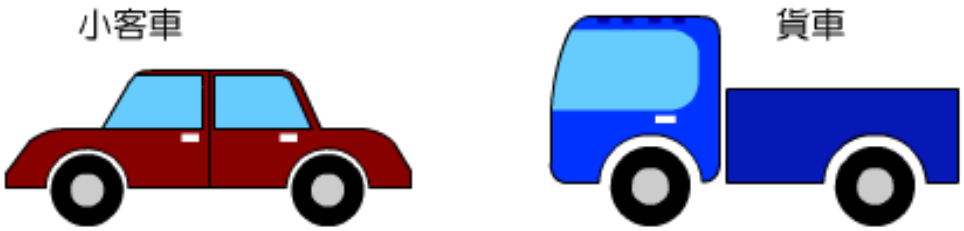
- 程式會從第一格開始讀取，並且由上而下執行，因此我們在撰寫程式的時候需要注意每個影格執行的順序，越需要先處理的東西，就要寫在越前面的影格（例如：各種變數的宣告，與函式的宣告）。
- 除了場景具備時間軸之外，每個影片片段（MovieClip）也都具備時間軸，某方面來講，其實場景也算某種特殊的MovieClip。

3-4物件導向

- AS3是一個「物件導向」的程式語言，而「物件導向」舉個最簡單的例子來說，大家可以把元件想像成一台車子，每台車都有很多性能值，比方他的最高時速是多少，他是幾人坐的等，這些資訊就是**屬性**，而車子也有很多功能，比方它可以載人也可以載貨物，這些功能就是**方法**。

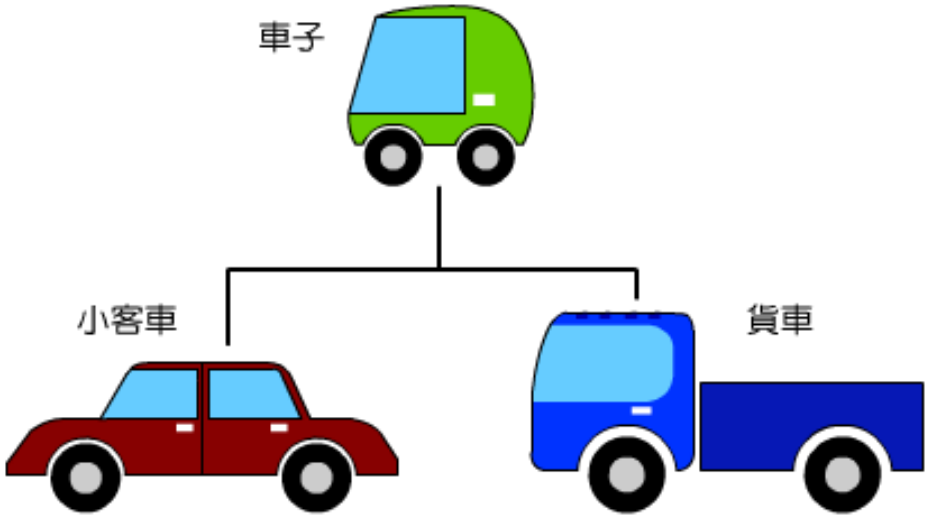
3-4物件導向

- 物件導向裡一個重要觀念就是「**類別**」，而所謂的類別就是幫物件分類，比方一個倉庫裡有許多車子，但是每台車子又有不同的種類，比方有的可能是小客車，有的則是貨車，我們就可以把它們分成不同的類別：像小客車類別、貨車類別...等。



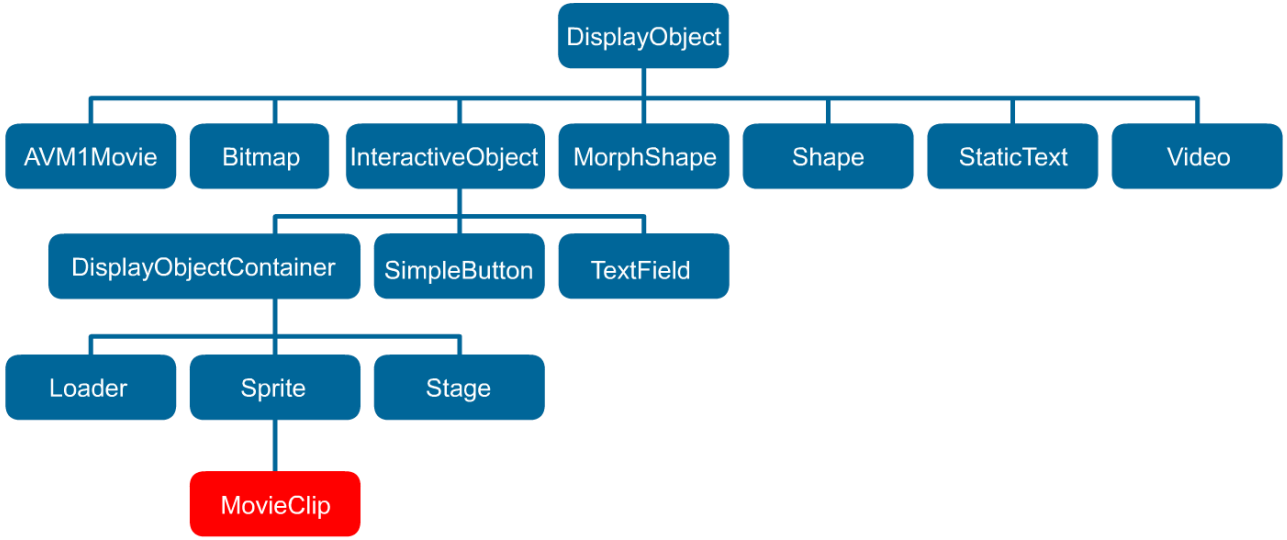
3-4物件導向

- 雖然分了不同的類別，但是我們可以發現不管是小客車或是貨車，基本的構造都是車子，所以其實還有一個最底層的類別是車子，而小客車跟貨車都是以車子為基礎再加以擴充性能的（小客車可能多加了座位，貨車則是多加了載貨的車廂），貨車不但擁有車子所有的性能，還多了載貨的功用，這個情況就稱為「**繼承**」。



3-5 Display Object類別

- 這張圖代表的是「Display Object」類別中所有的子類別，Flash中所有與顯示有關的物件都是繼承至「Display Object」類別。



程式碼解說：

3-6「Stage」類別

- 「Stage」類別代表的就是**場景**的類別，一個swf只會有一個「Stage」。

◎案例3-1：

程式碼：
1. trace("場景的寬度："+stage.stageWidth); 2. trace("場景的高度："+stage.stageHeight);

行數	程式碼的功用
1	使用「trace」指令輸出場景的寬度，這邊運用了「+」串聯前面的字串「場景的寬度：」與後面的「stage.stageWidth」資訊。
2	使用「trace」指令輸出場景的高度，這邊運用了「+」串聯前面的字串「場景的寬度：」與後面的「stage.stageHeight」資訊。

3-7 Sprite類別

- 當我們將場景上的東西建成影片片段時，這個影片片段就是一個物件，並且屬於MovieClip類別，而MovieClip類別式繼承另一個更底層的類別：「Sprite」。
- Sprite類別與MovieClip類別最大的差別就是它沒有時間軸，可以把它當作只有一個影格的MovieClip，而它一定要用程式宣告產生，無法從元件庫裡建立。

3-8「MovieClip」影片片段

- 屬性：

屬性	功用
x	元件在場景上的x座標，場景的左上角為座標（0,0）。
y	元件在場景上的y座標，場景的左上角為座標（0,0）。
name	元件在場景上的名稱（並非元件庫裡的名稱）。
width	元件的寬度。
height	元件的高度。
scaleX	元件的水平縮放比例，未縮放時是100，數字越大則放大越多。
scaleY	元件的垂直縮放比例，未縮放時是100，數字越大則放大越多。
totalFrames	元件中包含的總影格數。這個參數無法修改，只能讀取。
currentFrame	元件目前播放到哪一個影格。這個參數無法修改，只能讀取。
visible	元件是否能在場景上出現，是一個布林值，如果設定為false，則會無法在發佈之後的檔案看見，如果有元件內部有程式，也會無法運作。
alpha	元件的透明度，正常可見的情況是100，設定為最小值0時會呈現完全透明，但是雖然看不見，但是內部的程式還是可以運作。

3-8「MovieClip」影片片段

- 方法：

方法	功用
gotoAndStop(參數)	將播放磁頭移動到指定的影格，並停止。
gotoAndPlay(參數)	將播放磁頭移動到指定的影格，並播放。
stop()	讓元件停止播放。
play()	讓元件繼續播放。
addEventListener(參數1,參數2)	加入事件監聽，參數1是事件的類型，參數2是動作函式的名稱。
removeEventListener	移除事件監聽。
startDrag()	讓元件跟隨滑鼠指標移動。
startDrag()	解除startDrag造成的移動。

3-9影片片段的命名

- 用程式控制元件的第一步，就是幫**元件取名字**，因為有了名字，電腦才會知道程式設計師要操縱哪一個元件。接下來可以使用「.」語法來告訴元件要執行怎樣的指令。

3-10「.」語法

- 在AS3之中，「.」代表了元件名稱與元件屬性或函式指令的連接，結構如下：

  「元件名稱」.「元件屬性或函式指令」
- 假設場景上有一個元件叫做pig，將它的x屬性與y屬性重新存取。
- pix.x=0;
- pig.y=0;

3-10「.」語法

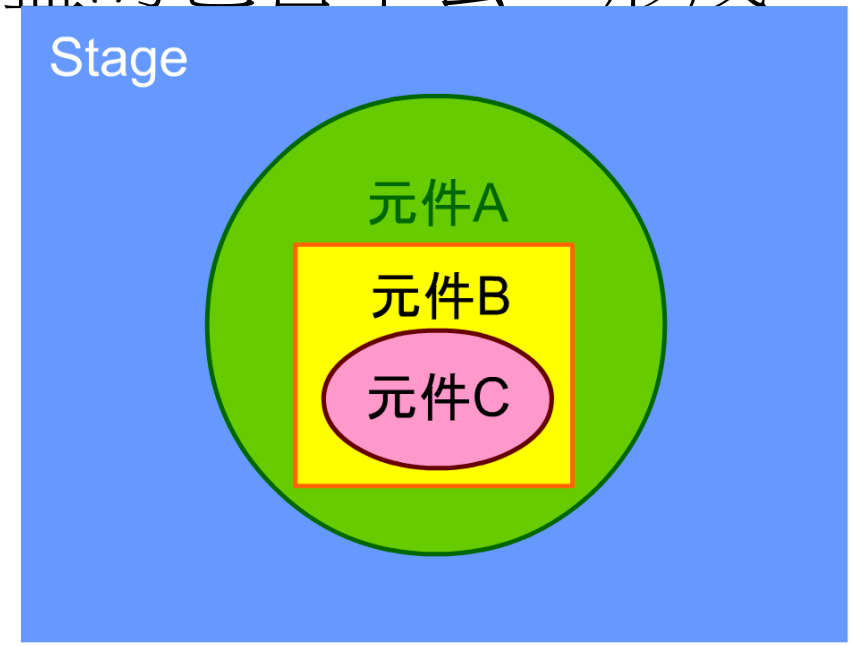
「元件名稱」.「函式指令」：

- 如果pig元件內部的第一格與第二格是不一樣的圖，而我們希望pig可以秀出並停在第二格的圖，我們可以寫下列指令控制元件。

  pig.gotoAndStop(2);

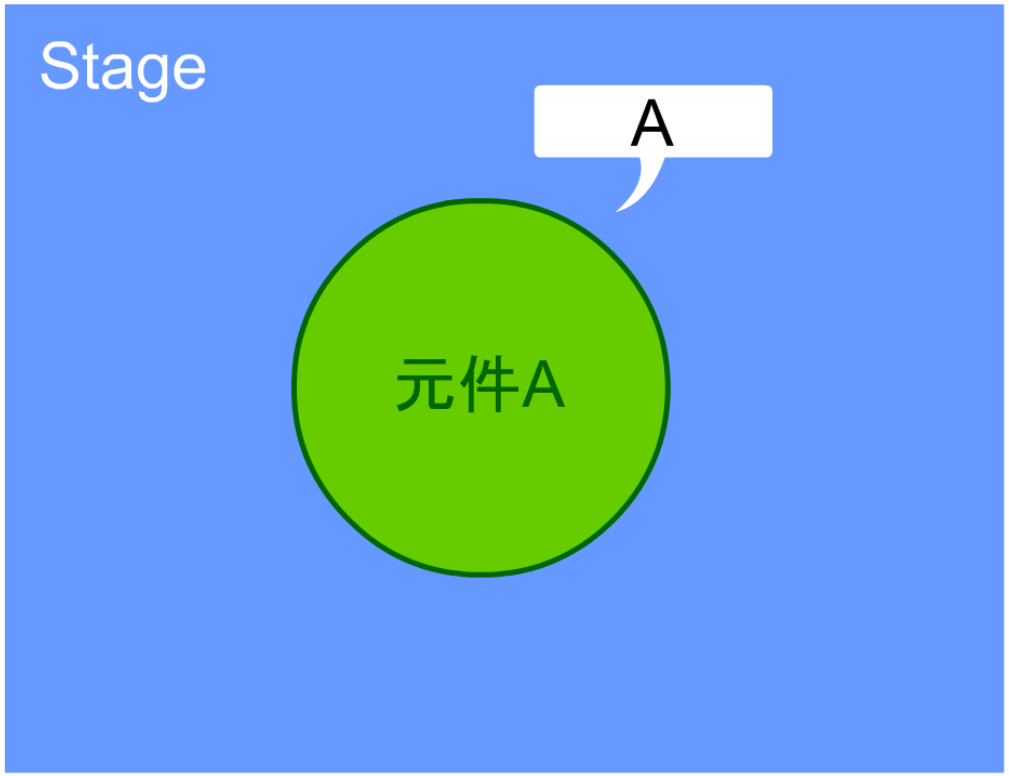
3-11 Flash的巢狀結構

- 每個Flash檔案都有一個場景（stage），而我們可以在場景上建立元件，而每個元件點進去，又可以擺放元件，元件與元件是可以無止盡的包含下去，形成一個巢狀的結構。



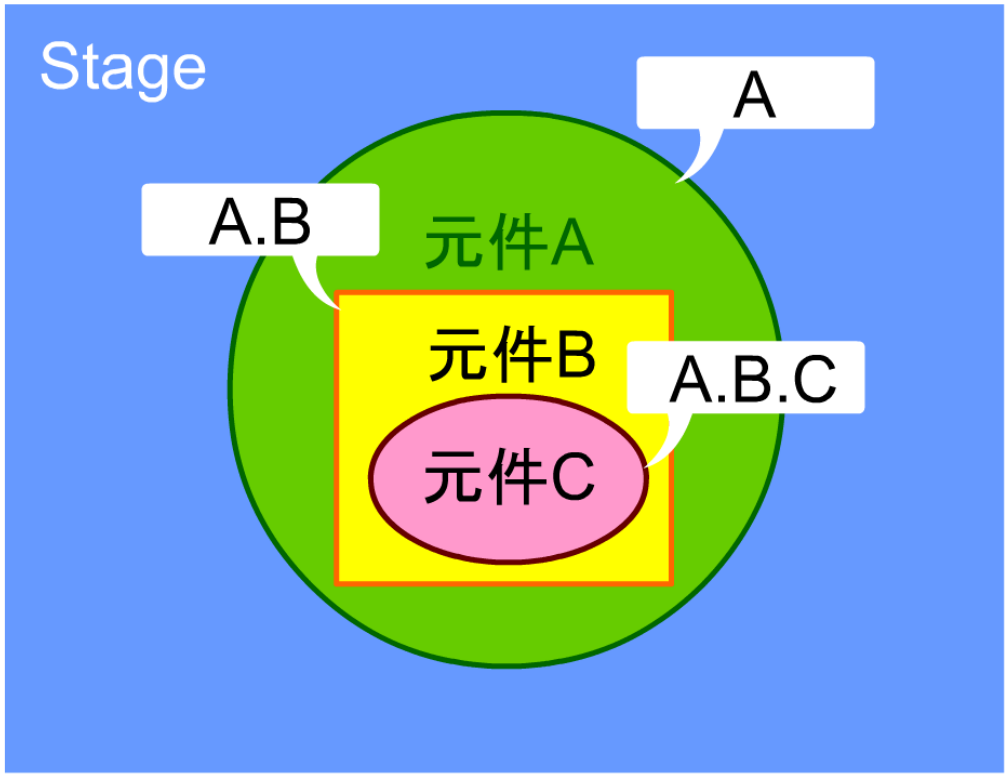
3-12 各個層級之間的關係

- 場景之下的元件



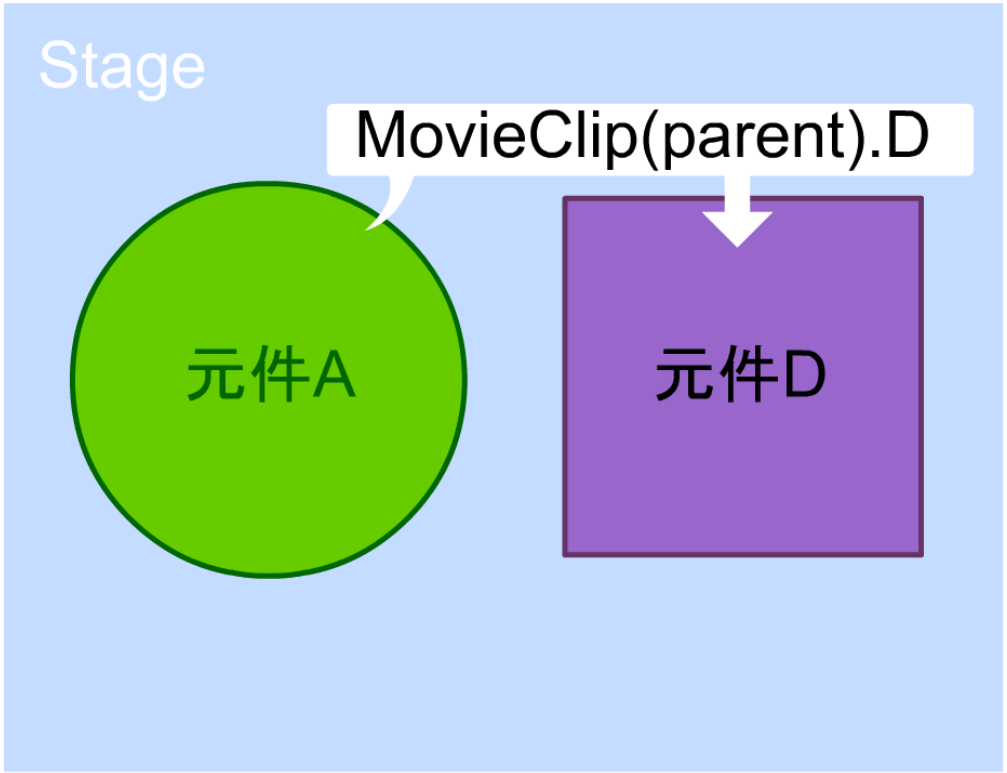
3-12 各個層級之間的關係:

- 包覆在元件「A」中的元件「B」。



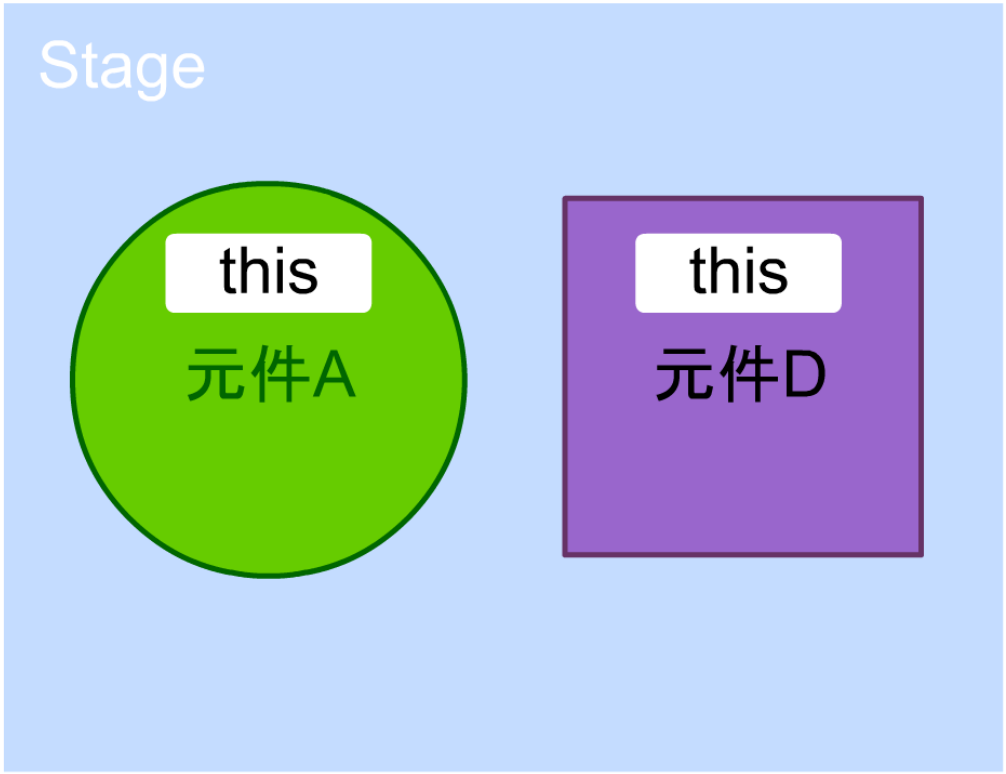
3-12各個層級之間的關係

- 從元件「A」中去呼叫元件「D」。



3-12各個層級之間的關係

- 在元件中呼叫自己



3-13章節案例：紙娃娃

◎目標：

讓使用者可以任意拖曳畫面上的物件來裝飾Mr.Pig。

◎程式原理：

利用影片片段的方法「startDrag」，讓使用者點選物件之後可以拖曳，再讓使用者放開滑鼠之後停止拖曳，本案例的程式碼雖然看似很長，但實際上是因為有四個物件而重複四次。

3-13章節案例：紙娃娃

程式碼

```
1.  //讓使用者可以拖曳帽子
2.  s1.addEventListener(MouseEvent.MOUSE_DOWN,s1down);
3.  s1.addEventListener(MouseEvent.MOUSE_UP,s1up);
4.
5.  function s1down(event:MouseEvent) {
6.          s1.startDrag();
7.  }
8.  function s1up(event:MouseEvent) {
9.          s1.stopDrag();
10. }
```

3-13章節案例：紙娃娃

行數	程式碼的功用
1	註解文字。
2	讓元件「s1」加入事件監聽，當它被點選時（滑鼠按下未放開），就會執行函式「s1down」中的動作。
3	讓元件「s1」加入事件監聽，當它被放開時（滑鼠放開），就會執行函式「s1up」中的動作。
5	函式「s1down」，讓使用者可以拖曳元件「s1」。
6	讓元件「s1」跟隨滑鼠移動。
8	函式「s1up」，讓使用者停止拖曳元件「s1」。
9	讓元件「s1」停止跟隨滑鼠移動。

3-13章節案例：紙娃娃

- 完成檔

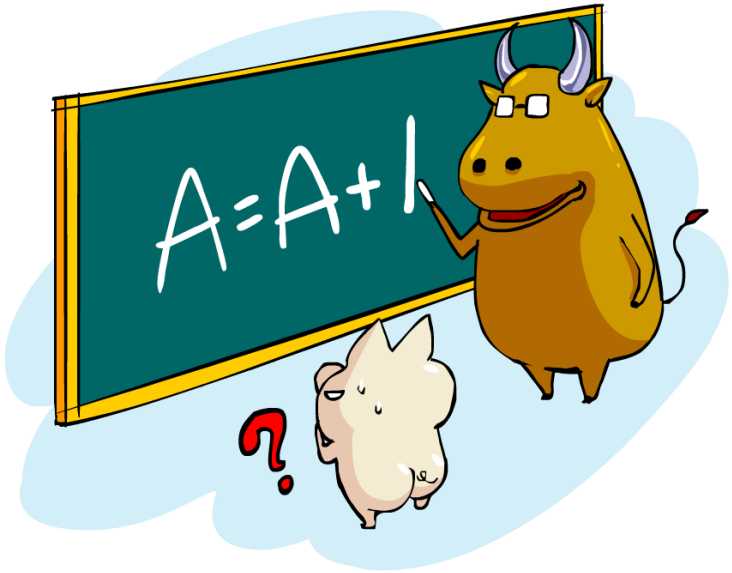
3-14習題

1. （　）**ActionScript3**中，程式碼只能寫在下列哪一種物件上？
（Ａ）關鍵影格（Ｂ）按鈕（Ｃ）影片片段（Ｄ）一般影格。
2. （　）下列哪一個不是物件？
（Ａ）MovieClip（Ｂ）Stage（Ｃ）Sprite（Ｄ）for。
3. （　）下列何者為非？
（Ａ）MovieClip具備時間軸（Ｂ）Sprite是繼承自MovieClip（Ｃ）Stage與MovieClip都是繼承自Display Object（Ｄ）Sprite就像只有一個影格的MovieClip。
4. （　）下列哪一個指令可以改變動畫播放的速度？
（Ａ）stage.frameRate（Ｂ）stage. scaleX（Ｃ）stage. scaleY（Ｄ）以上皆可。
5. （　）如果場景上有一個元件「**D**」，我們想從同在場景上的元件「**A**」裡去呼叫元件「**D**」，我們可以用下列哪一段語法？
（Ａ）parent.D（Ｂ）_parent.D（Ｃ）MovieClip(parent).D（Ｄ）MovieClip(parent.parent).D。

04-變數

4-1前言

- 在數學的世界裡，變數代表的是未知數，我們可以透過方程式去反向推測出變數的值，但在程式設計的世界裡，它不但是已知數，還是我們自己設定的數字。

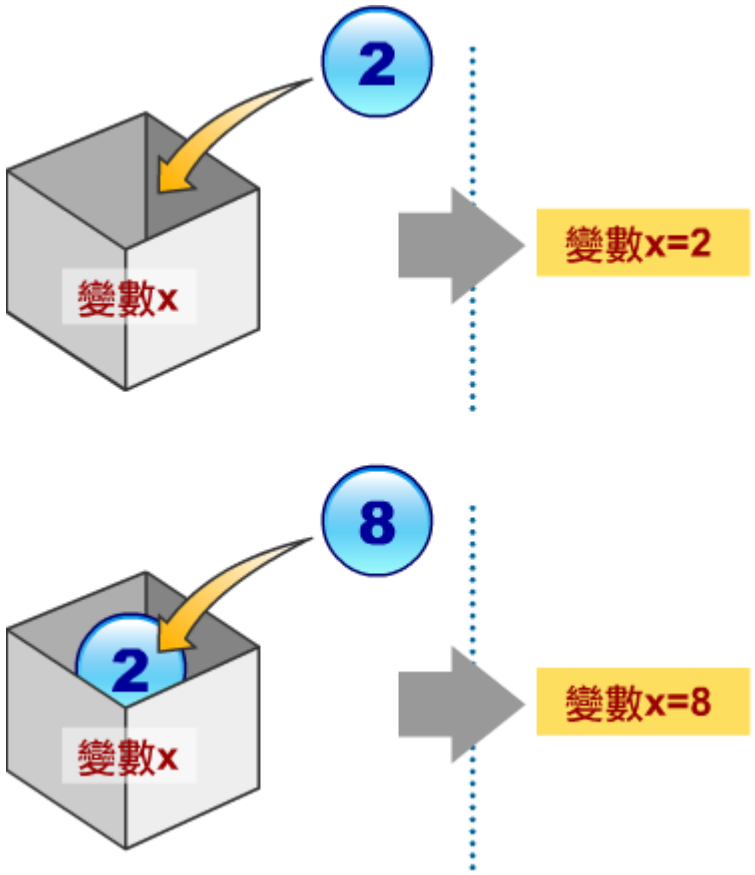


4-2學習目標

- 認識變數。
- 變數的種類。
- 變數的運算。

4-3何謂變數

- 變數就像是一個可以存放資料的盒子，而這個盒子可以替他任意取一個名字，但是這個盒子一次只能放一個東西，所以如果拿一個已經有放東西的盒子，再放進一個新的東西，舊的東西就會消失，換成新的東西。



4-4變數的宣告

- 宣告的語法如下：

  var 變數名稱:變數類型;

程式碼：
1. var box:int; 2. box=10; 3. trace(box);

4-4變數的宣告

※變數宣告的時候有一些規則要注意：

- 大小寫不同的字元，代表不一樣的變數（box與BOX電腦會判斷成不一樣的變數）。
- 某些系統使用的指令不能用在變數的命名（像trace就是）。
- 命名變數時，可使用英文加上數字，但是不可以只使用數字。

4-5等號的意義

- 「=」在AS程式設計裡並不是代表相等，而是代表儲存，也就是將新的東西放進盒子的動作。而電腦會先處理右邊的運算式，再把結果存進左邊的變數裡，所以利用這個原理，我們可以把變數裡的值修改之後再存放回同一個變數裡。

程式碼：

var box:int=10;
box=box+1;
trace(box);

4-6變數的類型

- 用盒子來裝東西的時候，東西有各種形狀，有的是長的，有的是寬的，所以需要不同的盒子來裝才能裝得下，而小東西如果準備一個大的盒子來裝，也會浪費很多空間。
- 宣告變數的類型的時候，替變數選擇最適合的類型，才能節省電腦記憶體的消耗，發揮最佳的效能，而某些特定的資料，也一定要選擇特殊的類型才能夠使用（像字串資料一定要宣告成String）。

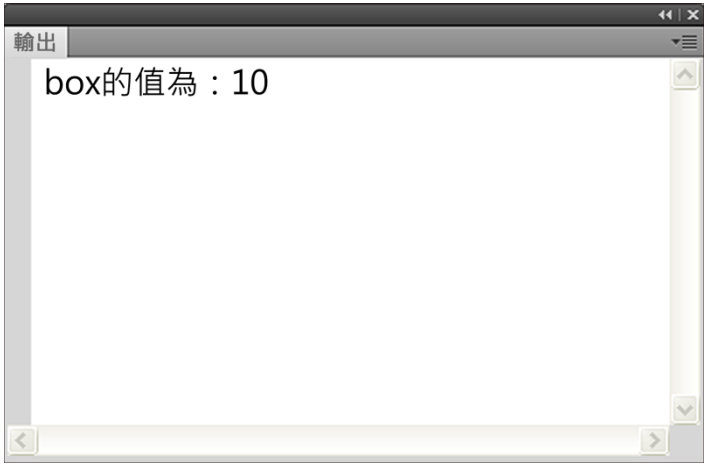
4-6變數的類型

變數類型	儲存的資料類型	預設值	說明
Boolean	true或false	false	布林值只能儲存兩個值，就是true或false，可以把它想像成對跟不對，當我們再做判斷式的時候常常會需要將結果回傳，如果結果只有對跟不對，就可以使用這個類型，它的好處就是占用的硬體資源少。
int	整數（包含正負）	0	可以儲存正負整數，但是不能儲存帶有小數的數字，通常整數資料會使用int類型而不使用Number類型，因為可以節省硬體資源的消耗。
Number	數字（包含小數）	NaN（Not a Number）	包含正負整數與小數的所有數字都能儲存，如果資料有小數就一定要宣告成Number，此外這個類型沒有數字大小儲存的上限。
uint	整數（不包含負數）	0	「無正負號」的整數，表示這個整數不能是負數
String	字串	null	可以儲存任何的字元（包含中文、英文與阿拉伯數字），但是字串無法做數學運算，所以需要做計算時必須要轉換為數字的類型。

4-7變數的輸出（存取值檢視）

- 變數在執行的過程中，其中的值也會不停地改變，所以我們會利用trace指令輸出變數的值，來確認程式是否依照我們所規劃的執行。

trace(“box的值為：”+box);



4-8除錯（Debug）

- 程式的運作都是依靠數學式的計算，各種資料在變數中的存取與修改之後，可以得到我們想要的結果。
- 跟人類的思考方式其實相差甚遠，所以我們在撰寫程式的時候，常常會發生執行結果與想像不同的情況。
- 除錯（Debug）或是除蟲，因為bug是蟲的意思，利用trace指令輸出變數中的值。

4-9 變數的運算

- 有兩個變數，a 與 b ：
- 加法：a+b
- 減法：a-b
- 乘法：a*b
- 除法：a/b(商，有小數)，a%b(只有餘數)

4-10章節案例：以按鈕控制的角色

◎目標：

使用按鈕控制場景上的元件上下左右移動。

◎程式原理：

我們藉由按鈕觸發，去修改元件的 x 屬性與 y 屬性，讓元件在場景上的位置改變，例如：我們今天要讓元件往右邊移動10畫素，就是讓元件原本的x屬性的值再增加10，依此類推。

4-10章節案例：以按鈕控制的角色

程式碼
1. //向上移動 2. function btnupf(event:MouseEvent) { 3. pig.y=pig.y-10; 4. } 5. btn_up.addEventListener(MouseEvent.CLICK,btnupf);

行數	程式碼的功用
1	註解文字。
2~4	函式「btnupf」，告訴按鈕「btn_up」當他被點選時，所要執行的動作：讓元件「pig」的y屬性減少10，因此場景上的元件就會往上移動。
5	讓按鈕「btn_up」加入事件監聽，當它被點選時，就會執行函式「btnupf」中的動作。

依此觀念完成剩下的三個按鈕。

4-10章節案例：以按鈕控制的角色

- 完成檔

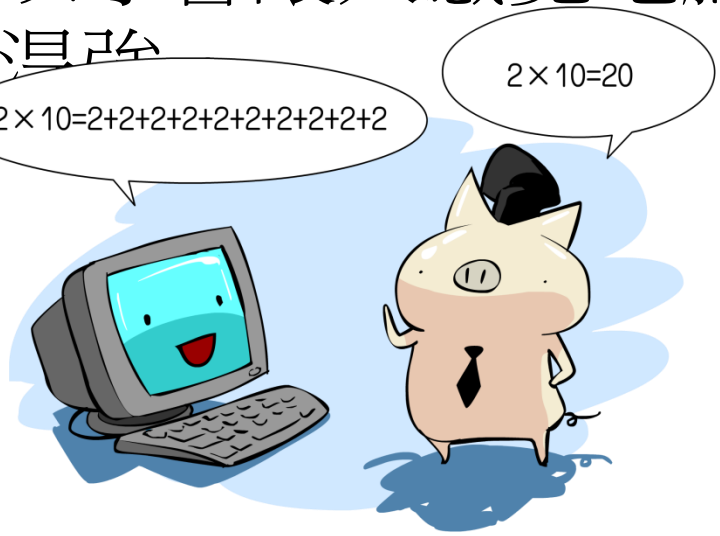
4-11習題

1. （　）下列哪一種變數類型代表正整數？
   （Ａ）Number（Ｂ）String（Ｃ）int（Ｄ）Boolean。
2. （　）「=」在程式設計中代表的意義為何？
   （Ａ）儲存（Ｂ）相等（Ｃ）絕對值（Ｄ）以上皆非。
3. （　）變數類型「Number」的預設值為何？
   （Ａ）0（Ｂ）NaN（Ｃ）false（Ｄ）null。
4. （　）a=9，b=3，a%b=？
   （Ａ）0（Ｂ）1（Ｃ）2（Ｄ）3。
5. （　）下列敘述的變數命名方式，何者為非？
   （Ａ）大小寫判定為不同的變數（Ｂ）不能使用系統指令當作變數名稱（Ｃ）可以只使用數字命名變數（Ｄ）變數命名時可以參雜英文與數字。

05-迴圈

5-1前言

- 電腦最強的能力，其實是在於可以快速而且持續重複不斷地做相同的事情，比方計算2x10，電腦是用2+2重複加10次，最後得到20的答案，只是電腦的加法速度遠遠超越人類，所以才會讓人感覺電腦的計算能力比人類來得強

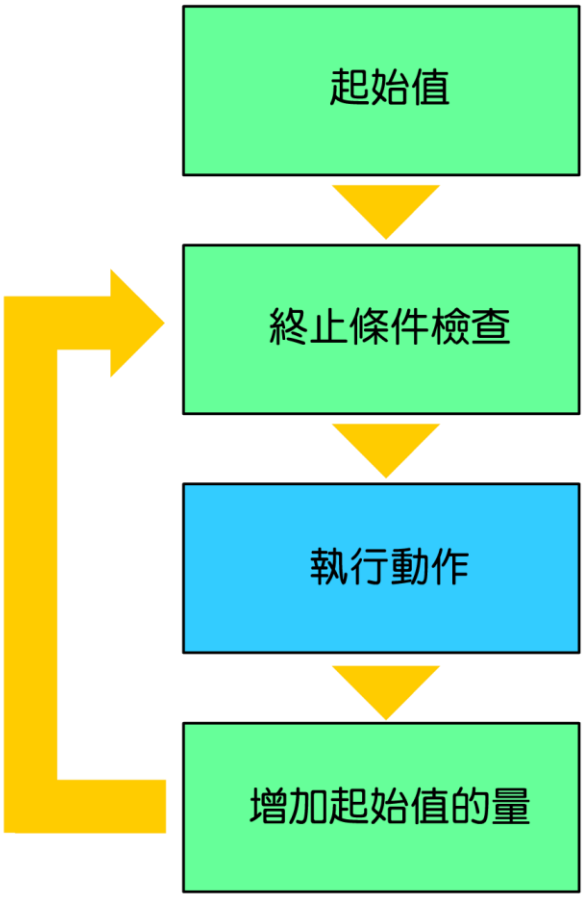


5-2 學習目標

- 了解迴圈的使用方式。
- 認識ActionScript3中各種迴圈語法。
- 認識文字類別。

5-3迴圈的概念

- 迴圈的基本構成要素有三點：
- 起始值
- 終止條件
- 每執行一回合，增加的量（會決定這個迴圈要跑幾次）



5-4迴圈的種類

迴圈的類型	特色	使用時機
for	最常用也最常見的迴圈，它的優點在於宣告時會把所有需要用到的參數都寫在第一行，方便我們檢查跟撰寫。	通常我們會在知道執行次數的情況下，使用for迴圈，比方我們要讓一個值輸出十次，我們就可以使用for迴圈。
while	屬於比較簡化的迴圈，它只需要設定終止條件。起始值與增加的量都得另外寫在程式碼之中。	while迴圈因為只能設定終止條件，所以大多被用在不知道需要執行幾次的情況下，比方我們
影格迴圈	利用Flash影格的特性來撰寫的迴圈，需要搭配時間軸控制程式與判斷式一起使用，跑迴圈的時間會跟隨影片的影格速率改變。	影格迴圈因為時間軸的特性，所以能夠抓出執行時間，我們通常用來製作計時器。

5-5迴圈的寫法

- for迴圈：

```
for(起始值; 終止條件; 每執行一回合，增加的量){
    需要重複執行的程式碼
}
```

5-5迴圈的寫法

- 計算10的階乘

程式碼：
1. var answer:int=1; 2. for(var i:int=1;i<=10;i++){ 3. answer=answer*i; 4. } 5. trace(answer);

行數	程式碼的功用
1	宣告一個變數來儲存階乘的答案。
2	設定迴圈的條件，起始值為var i:int=1，終止條件為i<=10，每回合增加的量為i++（是i=i+1的簡寫，表示每跑一輪，i就多1）。
3	變數「answer」每執行一輪，就會累積上一輪的乘積。
5	輸出答案（注意輸出的指令寫在迴圈之外，因為我們要的是最終的答案）。

5-5迴圈的寫法

執行回合數	answer=answer*i
1	1=1*1
2	2=1*2
3	6=2*3
4	24=6*4
5	120=24*5
6	720=120*6
7	5040=720*7
8	40320=5040*8
9	362880=40320*9
10	3628800=362880*10

5-5迴圈的寫法

- while迴圈：

宣告起始值

whiile(終止條件){

需要重複執行的程式碼

每執行一回合，增加的量

}

5-5迴圈的寫法

- 計算10的階乘

程式碼：

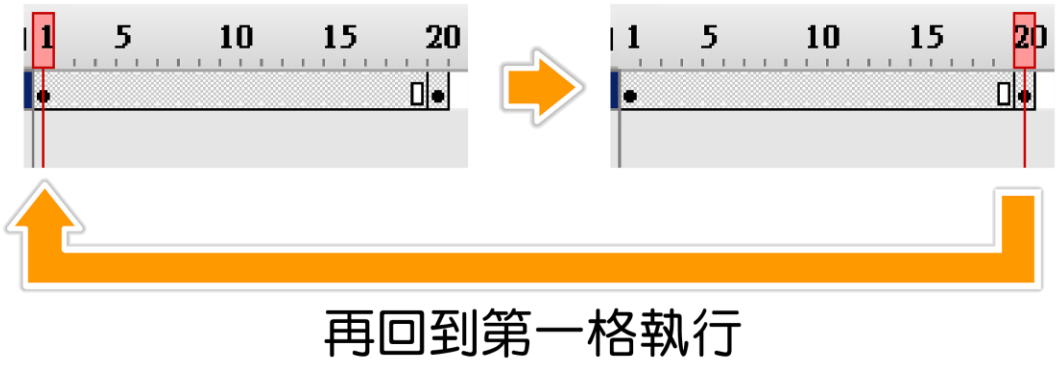
```
1. var answer:int=1;
2. var i:int=1;
3. while(i<=10){
4.    answer=answer*i;
5.    i=i+1;
6. }
7. trace(answer);
```

行數	程式碼的功用
1	宣告一個變數來儲存階乘的答案。
2	宣告一個變數當作while迴圈的起始值。
3	設定迴圈的條件，終止條件為i<=10。
4	變數「answer」每執行一輪，就會累積上一輪的乘積。
5	設定每回合增加的量。
7	輸出答案（注意輸出的指令寫在迴圈之外，因為我們要的是最終的答案）。

5-5迴圈的寫法

- 影格迴圈：

  Flash影格的特性就是會輪播，每當播到最後一格的時候，又會從第一格開始播，利用這種循環的特性，我們也可以寫成迴圈，但是通常我們在使用的時候，還會搭配判斷式。



5-6中止迴圈執行（break）

- 使用「break」指令讓迴圈停止執行，讓迴圈停止執行的好處就是可以避免系統資源的浪費。
- 通常會搭配第五章所要教的判斷式(if…else…)。

5-6中止迴圈執行（break）

- 用迴圈判斷數字259是否為質數

程式碼：

```
1. var num:int=259;
2. var answer:String="是質數";
3. for(var i:int=2;i<num;i++){
4.   if(num%i==0){
5.     answer ="不是質數";
6.     break;
7.   }
8. }
9. trace(answer);
```

5-6中止迴圈執行（break）

行數	程式碼的功用
1	宣告一個變數來儲存我們要計算是不是質數的數字。
2	宣告一個變數儲存答案，預設為「是質數」，如果不是的話，會在判斷式中修改答案。
3	設定迴圈，跳過1與它本身，只要有數字能整除它，就代表它不是質數。
4	判斷式，如果餘數為0（代表整除，「%」符號與除號的功能相同，只是除出來的答案是兩數相除之後的餘數），就執行下列的程式碼。
5	將答案修改為「不是質數」，因為已經找到因數可以整除該數字。
6	中止迴圈。
9	輸出答案。

5-7雙重迴圈

- 許多狀況下我們需要使用雙重迴圈（像本章節案例的九九乘法表），也就是一個迴圈中在包覆另外一個迴圈。

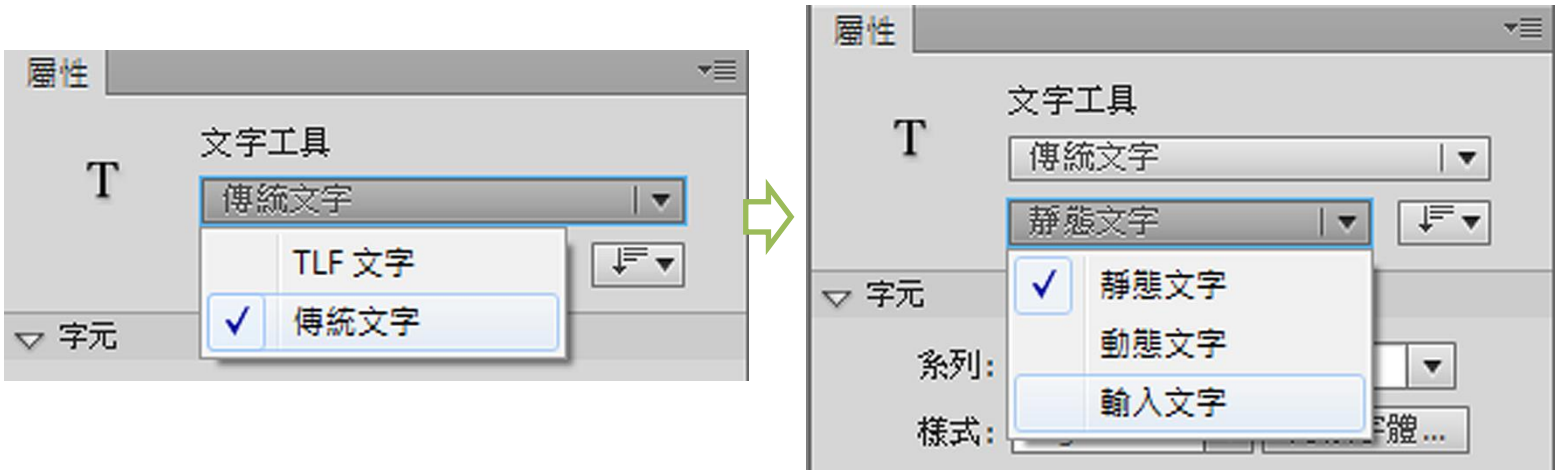
```
for(var i:int;i<10;i++){
    for (var j:int=0; j<i; j++) {
    }
}
```

第一層迴圈

第二層迴圈

5-8文字

- 在Flash之中的文字有分**傳統文字**與**TLF文字**，而我們要介紹的部分是傳統文字。
- 傳統文字一共可以分成三種類型，分別是靜態文字、動態文字、和輸入文字，我們可以從屬性欄位中確認文字欄位的性質，除了靜態文字之外，動態文字與輸入文字都是要跟程式搭配使用。



5-8文字

文字屬性	說明
靜態文字	美術製作時所使用的一般文字，無法用程式控制。
動態文字	可以使用程式將資料呈現在場景上的文字框，但是不會隨著文字多寡自動調整框架大小，因此我們在製作時必須要注意輸出文字的多寡，預先設置好大小。
輸入文字	讓使用者透過鍵盤輸入文字的欄位。

5-9用程式控制動態文字

- 首先利用工具列的文字工具在場景上放置一個文字框，將它的屬性設定為動態文字，這時候我們就會發現多了一個欄位可以替它取名字，我們將這個文字框命名為「showTXT」。

showTXT.text=”我們要顯示的字串”;

5-9用程式控制動態文字

- 用程式產生動態文字，所有的文字都屬於TextField類別。

程式碼：
1. var showTXT: TextField=new TextField(); 2. showTXT.text="我們要顯示的字串"; 3. addChild(showTXT);

行數	程式碼的功用
1	宣告一個TextField類別的實體（就像我們放置的文字框）。
2	將顯示的字串加入「text」屬性。
3	將TextField的實體加到場景顯示（關於addChild可以參閱第十一章）。

5-9用程式控制動態文字

- TextField類別也提供了很多屬性與方法，但是常用的設定，我們都可以在屬性面板中調整，因此建議大家可以先把想要呈現的文字顏色或行距等…都先調整好，就可以不用程式去修改TextField底下的屬性。

5-10 TextFormat類別：文字樣式改變

- TextFormat就像是網頁裡使用的CSS，可以設定一些文字編排的參數，在套用在我們的文字框裡，使用TextFormat時，必須先宣告一個TextFormat物件。

var TXF=new TextFormat(“參數”);

5-10 TextFormat類別：文字樣式改變

- TextFormat可以設定的參數：

TextFormat(font:String = null, size:Object = null,
color:Object = null, bold:Object = null,
italic:Object = null, underline:Object = null,
url:String = null, target:String = null,
align:String = null, leftMargin:Object = null,
rightMargin:Object = null, indent:Object = null,
leading:Object = null)

5-10 TextFormat類別：文字樣式改變

參數名稱	資料類型	說明
font	字串	文字的字型。
size	數字	文字的大小。
color	字串	文字的顏色(例如：0x000000代俵黑色)。
bold	布林	是否為粗體(true代表是，false代表否)。
italic	布林	是否為斜體(true代表是，false代表否)。
underline	布林	是否有底線(true代表是，false代表否)。
url	字串	超連結的網址。
target	字串	設定超連結是否要新開視窗，_self是在本身的視窗開啟，_blank是在新的視窗中開啟。
align	字串	段落對齊，left為對齊左邊，middle為對齊中間，right為對齊右邊。
leftMargin	數字	段落的左方邊界，以像素為單位。
rightMargin	數字	段落的右方邊界，以像素為單位。
indent	數字	指出段落中從左方邊界到第一個字元的縮排。
leading	數字	行距。

5-10 TextFormat類別：文字樣式改變

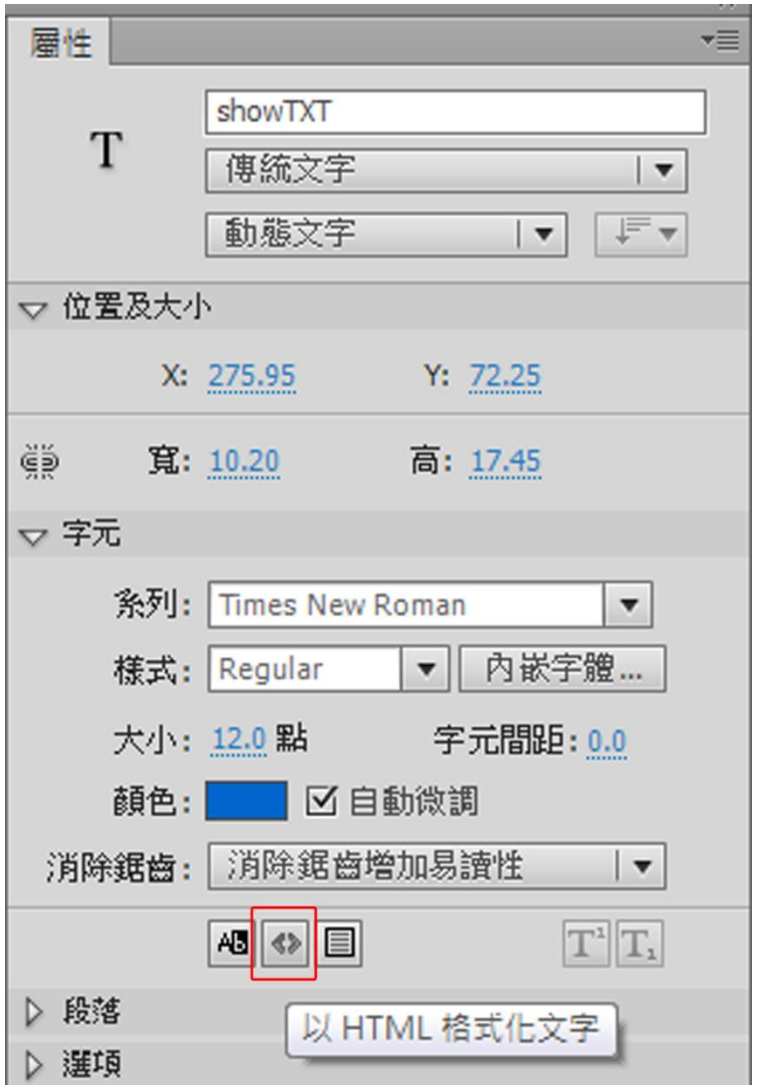
- 將文字的大小設定為30像素

程式碼:
1. var showTXT: TextField=new TextField(); 2. showTXT.text="我們要顯示的字串"; 3. var TXF=new TextFormat("",30); 4. showTXT.setTextFormat(TXF): 5. addChild(showTXT);

行數	程式碼的功用
1	宣告一個TextField類別的實體（就像我們放置的文字框）。
2	將顯示的字串加入「text」屬性。
3	將字元大小設定為30像素。
4	使用setTextFormat指令，將這些設定好的格式套入文字欄位。
5	將TextField的實體加到場景顯示（關於addChild可以參閱第十一章）

5-11 HTML語法套用

- 我們可以直接從屬性面板中勾選「以HTML格式化文字」。欄位中直接輸入HTML語法，發佈之後會自動將它轉換成網頁的形式。



5-11 HTML語法套用

- 以程式碼的方式來寫:

  動態文字.htmlText="Html的內容";

- 目前支援的Html標籤：

錨點標籤 <a>	影像標籤 <img>	Span 標籤
粗體標籤 <b>	斜體標籤 <i>	文字格式標籤 <textformat>
換行標籤 	清單項目標籤 	底線標籤 <u>
字體標籤 <font>	段落標籤 <p>	

5-8章節案例：九九乘法表

◎目標：

讓電腦幫我們產生九九乘法表裡的數字，並且排列出來。

◎程式原理：

九九乘法表的概念就是，就是先從1開始，一直乘到9，再換到2，在乘到9...依此類推，我們可以發現1x1=1、1x2=2...乘號左邊的數字一直都沒改變，直到右邊的數字變成9，才會加1，變成2，因此我們就可以使用兩個迴圈來模擬這種情況，第一個迴圈跑左邊的數字，第二個迴圈跑右邊的數字。

5-8章節案例：九九乘法表

◎程式原理：

變數「i」	迴圈內的程式執行	
i=1	變數「j」	i*j
	j=1 j=2 j=3 j=4 j=5 j=6 j=7 j=8 j=9	1*1=1 1*2=2 1*3=3 1*4=4 1*5=5 1*6=6 1*7=7 1*8=8 1*9=9

5-8章節案例：九九乘法表

- 用文字工具在場景上拖曳出一個範圍，並且在屬性面板設定為動態文字。
- 將文字框命名為showTXT。

5-8章節案例：九九乘法表

程式碼
1. //建立一個空字串，用來存放最後顯示在畫面上的結果 2. var AllStr:String=" ";

行數	程式碼的功用
1	註解文字。
2	建立一個變數AllStr，它的型態是字串，並且存入空的字串（兩個”中間是沒有任何東西的）。它的功用是儲存最後要輸出的結果。

5-8章節案例：九九乘法表

程式碼

4. //製造出99乘法表
5. for (var i:int=1; i<=9; i++) {
6. for (var j:int=1; j<=9; j++) {
7. AllStr=AllStr+i*j+" ";
8. }
9. //\n代表換行符號，不會被顯示出來
10. AllStr=AllStr+"\n";
11. }

行數	程式碼的功用
5	第一個迴圈，代表乘號左邊的數字，變數i每執行一輪就會增加1，直到9為止，每一輪都會重新執行第二個迴圈裡的程式，因此第一輪結束時，就會產生出1×1=1...1×9=9的結果。
6	第二個迴圈，代表乘號右邊的數字，j每執行一輪就會增加1，直到9為止。
7	將i*j的結果存入變數AllStr，由於AllStr是字串，所以i*j結果也會被判定為字串，因此會一直累加在字串後面，多加的空白字元，是為了區隔數字，避免數字都連結在一起。
10	當執行完第二個迴圈，就代表已經完成一排數字，所以要插入一個換行符號，讓輸出的資料換下一行，這個\n並不會被顯示出來。

5-8章節案例：九九乘法表

程式碼
13.//將最後的結果顯示在動態文字框 14.showTXT.text=AllStr;

行數	程式碼的功用
14	電腦已經把九九乘法表裡的資料都計算出來並且存在變數AllStr之中，我們透過動態文字框showTXT來顯示出結果。

5-8章節案例：九九乘法表

- 完成檔

5-9習題

1. （　）下列哪一種迴圈只需要終止條件？
（A）while（B）for（C）for…in（D）以上皆非。
2. （　）以下關於for迴圈的敘述何者為非？
（A）最常使用的迴圈（B）不需要宣告終止條件（C）需要宣告起始值（D）需要宣告每回合增加的量。
3. （　）關於迴圈中下「break」指令的功用，何者為非？
（A）中斷迴圈的執行（B）節省系統的資源（C）無法用在while迴圈（D）通常會搭配判斷式一起使用。
4. （　）一個正常的迴圈的必要的條件下列何者為非？
（A）起始值（B）終止條件（C）每回合增加的量（D）判斷式。
5. （　）下列關於九九乘法表的程式中，哪個敘述是不對的？
（A）需要使用兩個迴圈（B）有兩個起始的變數（C）不能使用while迴圈來寫（D）內部的迴圈的起始值每回合會根據外部的迴圈改變。

06-判斷式

6-1 前言

- 判斷式是電腦最接近人類的一種程式機制，它其實很簡單，用口語來講，就是「如果...那就這樣，如果不是...那就那樣」。

6-1 前言

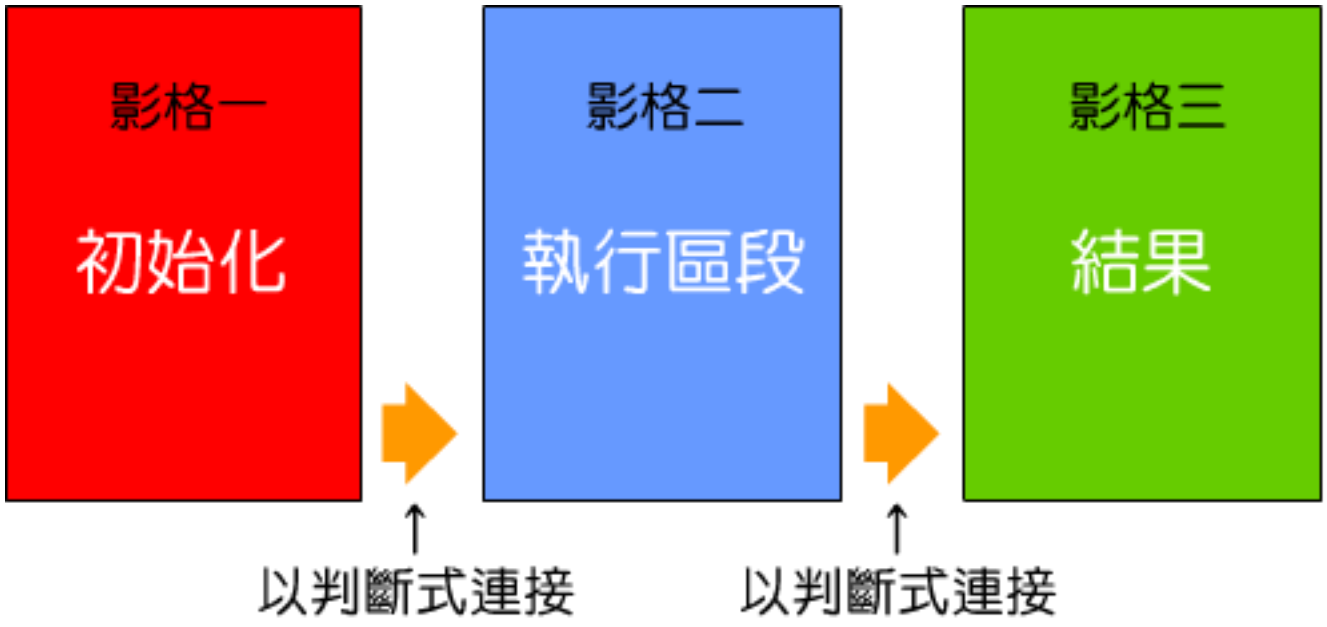
- 一個「=」號在程式裡代表儲存，兩個「=」號「==」才代表相等。
- 「Math」類別，它是在AS3中少數不用宣告就可以使用的類別，它代表了許多數學常數。

6-2學習目標

- 認識判斷式的種類。
- 了解流程控制。
- 特殊類別「Math」。

6-3判斷式等於流程控制

- 決定程式從目前的狀態跳到下一個流程，依靠的就是判斷式，所以說判斷式可以被看作是每個流程區塊間的橋樑。



6-3判斷式等於流程控制

- 常用的判斷式

判斷式語法	可輸入的條件	特色
if…else	無限制	「if」代表的就是「如果…」，而「else則代表「其它所有不是if的情況」，「if…else」判斷式最大的好處就是可以設定許多條件，要全部都符合才能執行。
switch	只有一個條件	「switch」只針對一個變數的內容作判斷，如果變數的值符合某個條件，就會執行特定的動作，主要使用在一個變數需要判斷大量資料的情況（比方：這個變數從1到100都有對應的動作，就可以使用「switch」），對於程式的優化也有好處。

6-4「if...else」判斷式

- 「if...else」判斷式的語法如下：

```
if(判斷條件){
    執行動作1
}else if{
    執行動作2
}else{
    執行動作3
}
```

6-5 「switch」判斷式

- 「switch」判斷式的語法如下：

```
switch(被拿來判斷的變數){
 case 條件1 :
   執行動作1
   break;
 case 條件2 :
   執行動作2
   break;
 case 條件3 :
  …依此類推
default:  如果條件都不合，執行這個動作
  break;
}
```

6-6「&&」與「||」的特殊判斷

- 邏輯運算子「&&（AND）」與「||（OR）」可以參考下列的表格說明：

邏輯運算子	代表意義	符合的情況
&&	AND(...和...都符合)	每個條件式都要完全相等才能執行動作。
||	OR(...或...其中一項符合)	所有條件式中，只要有一個條件符合，即可執行動作。

6-7 布林 (true or false)

- 在宣告變數時有一種特殊的類型叫「Boolean」，也就是所謂的**布林**，這種類型的變數只有兩種值，一個是**true**代表正確，另外一個是**false**代表錯誤（false為預設值），通常我們宣告布林變數都是用來搭配判斷式來使用。

6-8「Math」類別－數學運算類別

- AS將這些方法全都整合在「Math」類別之下，而「Math」類別是一個很特別的類別，它不需要宣告就可以使用，舉例來說下列的程式碼會幫我們計算出平方根：

```
trace(Math.sqrt(4));
```

輸出結果：2

6-9 亂數（random）

- 亂數就是沒辦法預測的數字，在電腦遊戲中代表很重要的意義，我們在玩紙上遊戲時（像大富翁），也會使用像骰子之類的道具，來製造不可預測結果，增加遊戲的樂趣。
- 「Math.random()」的時候，它會回傳一個隨機的數字，而這個數字是一個小於1但是大於或等於0的小數，因此我們如果要用使用整數的亂數時，我們就會配合「Math.floor()」的方法，來找出整數的亂數。

6-9 亂數（random）

- 隨機回傳從0到5的亂數：

程式碼：
1. var ranNum:int; 2. ranNum= Math.floor(Math.random() *6); 3. trace(ranNum);

行數	程式碼的功用
1	宣告一個變數來儲存回傳的數字。
2	將亂數乘上6之後再取得最接近的整數。
3	輸出答案。

6-10章節案例：猜拳遊戲

◎目標：

讓使用者與電腦猜拳的小遊戲。

◎程式原理：

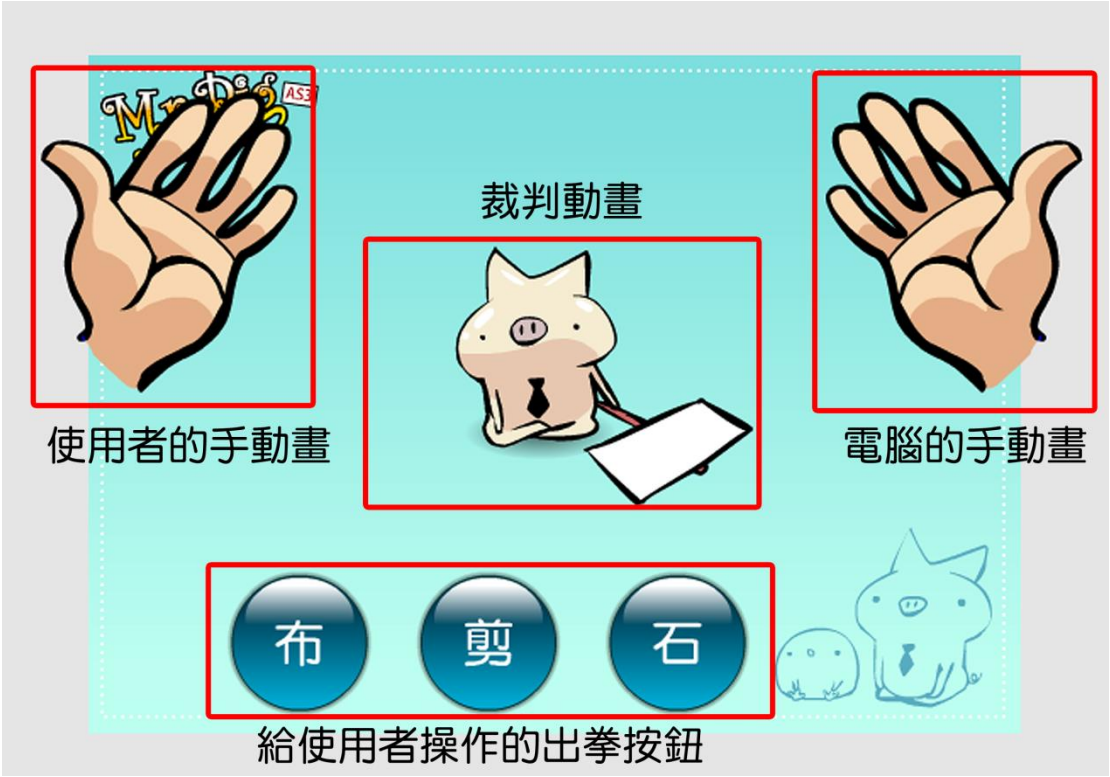
程式分為「決定要出什麼拳」與「判斷勝負」，因此我們可以使用兩個影格來製作，而我們分別用一個數字來代表剪刀、石頭、布這三種出拳，我們都知道剪刀石頭布的規則，是剪刀勝布、布勝石頭、石頭勝剪刀，我們用1代表布，2代表剪刀，3代表石頭，因此只要兩個相減，等於1或是-2就代表是我們勝利，除此之外都代表我們輸或是平手。可以參考以下這張表格：

6-10章節案例：猜拳遊戲

玩家	電腦	勝負(兩個相減的結果)
布(1)	布(1)	平手(0)
	剪刀(2)	失敗(-1)
	石頭(3)	勝利(-2)
剪刀(2)	布(1)	勝利(1)
	剪刀(2)	平手(0)
	石頭(3)	失敗(-1)
石頭(3)	布(1)	失敗(2)
	剪刀(2)	勝利(1)
	石頭(3)	平手(0)

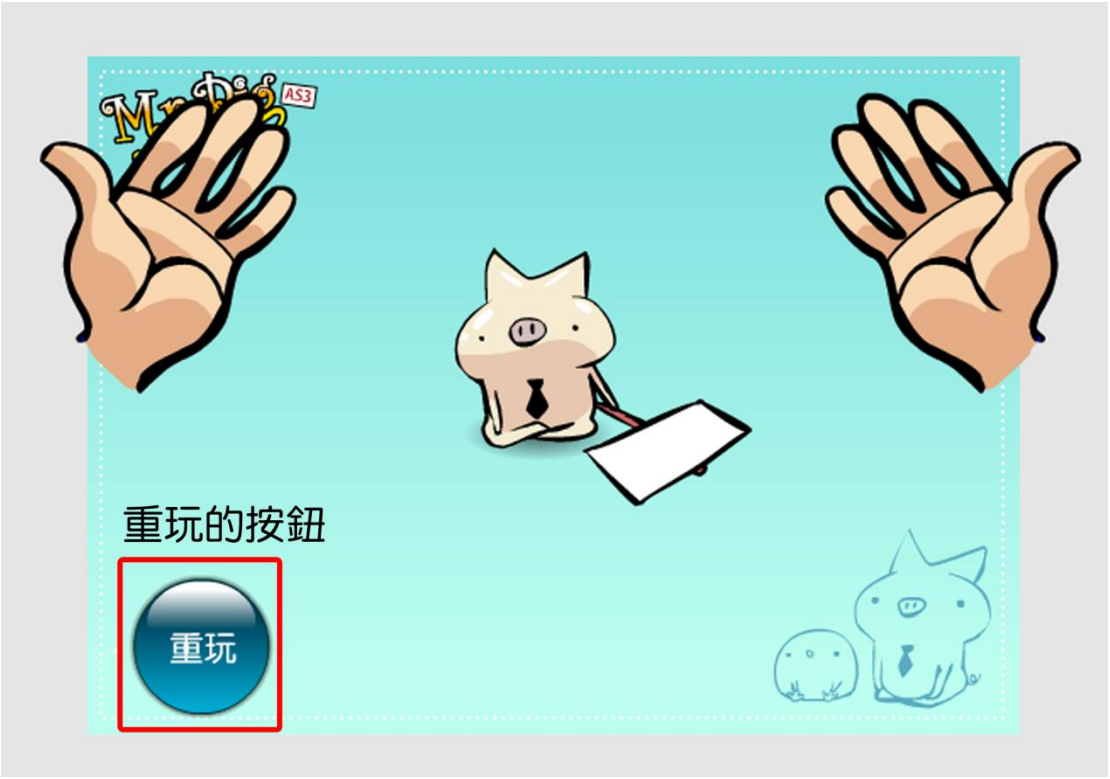
6-10章節案例：猜拳遊戲

- 影格1(標籤gamestart)



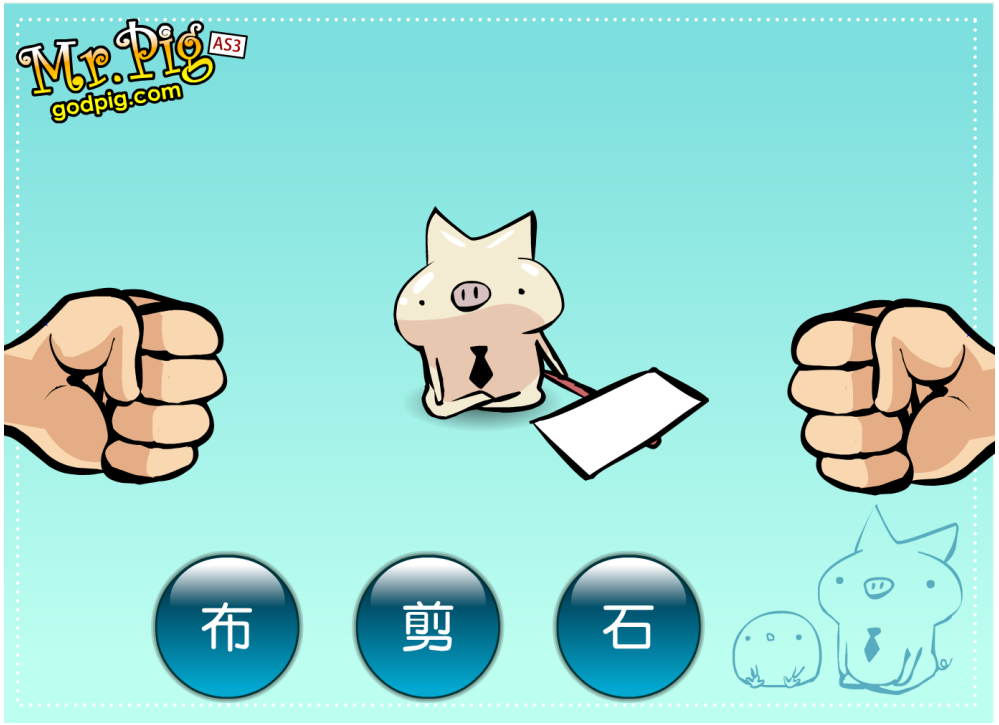
6-10章節案例：猜拳遊戲

- 影格2(標籤gameend)



6-10章節案例：猜拳遊戲

- 完成品



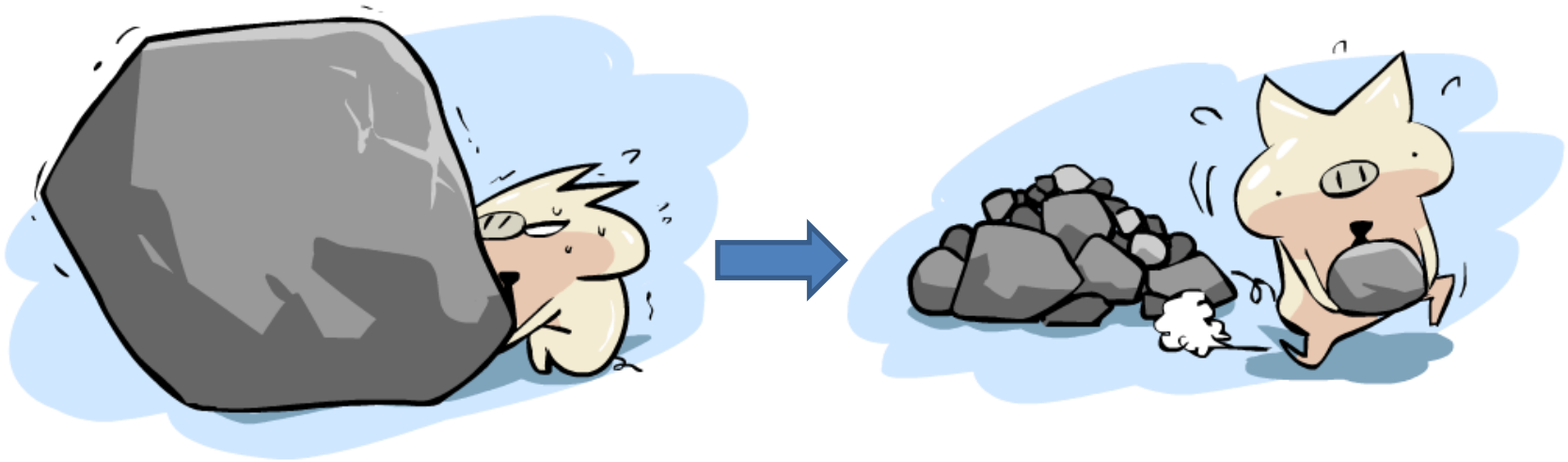
6-11習題

1. （　）「**if…else**」判斷式最多可以有幾個判斷條件？
（Ａ）一個（Ｂ）兩個（Ｃ）三個（Ｄ）無限多個。
2. （　）關於「**switch**」判斷式的敘述何者為非？
（Ａ）只能有一個判斷條件（Ｂ）每個case中要有一個break（　）所有case都不成立時會執行default（Ｄ）可以使用「&&」做判斷。
3. （　）變數類型「**Boolean**」的預設值為何？
（Ａ）0（Ｂ）NaN（Ｃ）false（Ｄ）null。
4. （　）關於「**Math**」類別的敘述何者為是？
（Ａ）不需要宣告就可以使用（Ｂ）Math 類別沒有屬性（Ｃ）Math 類別沒有方法（Ｄ）Math 類別並不是一個物件。
5. （　）「**X=**Math.random();」，則X有可能為？
（Ａ）1（Ｂ）2（Ｃ）0（Ｄ）-1。

07-函式編寫

7-1 前言

- 當我們遇到一個大問題的時候，可以先細分成比較小的問題，各別去處理它，這就是函式的概念。

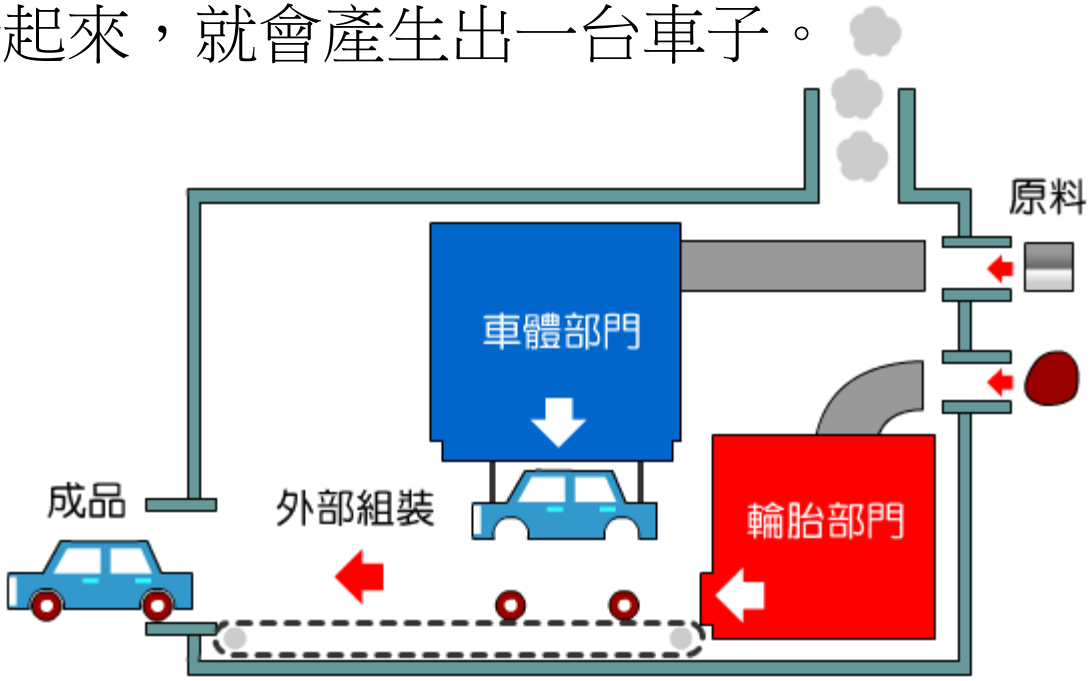


7-2學習目標

- 了解函式的架構
- 函式在ActionScript3.0中扮演的角色
- 變數在函式裡的變化
- 事件與函式之間的關連

7-3化大為小，化繁為簡

- 把整個大程式當作是一個生產汽車的大工廠，而函式就是工廠中的各個零件部門，生產輪胎、座椅...等等零件，而生產這些零件我們需要原料（參數），所以把原料交給零件部門之後，就會產生零件（回傳值），最後再將這些零件組合起來，就會產生出一台車子。



7-4函式的宣告

- 函式的宣告方式如下：

```
function 函式名稱(參數1,參數2...){
執行動作
return 回傳值
}
```

7-4函式的宣告

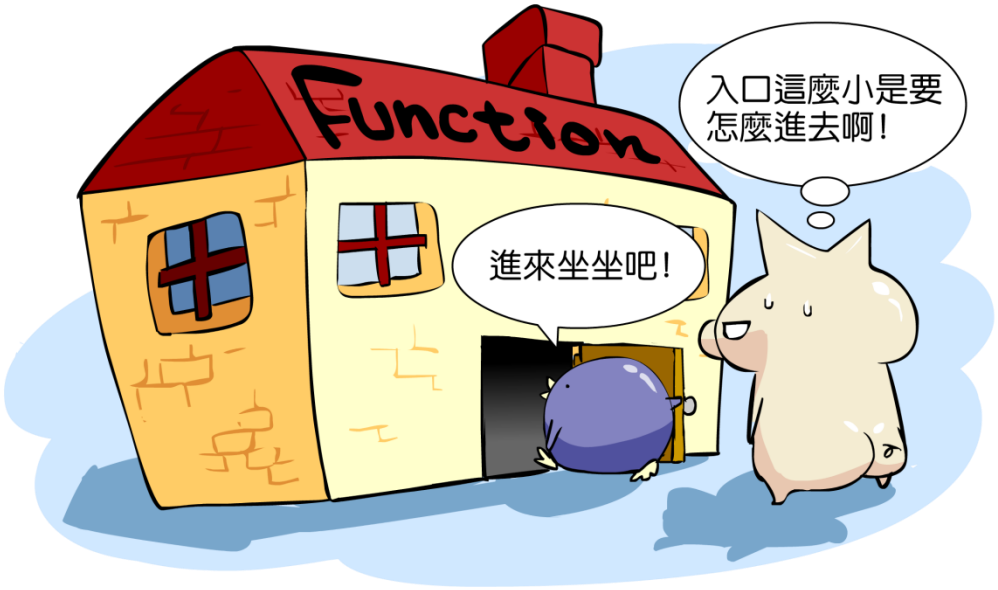
- 把兩個傳進去的參數相加之後傳回結果

程式碼：
1. function Sum(a:int,b:int){ 2. return a+b; 3. } 4. trace(Sum(10,20));

行數	程式碼的功用
1	宣告一個函式「Sum」，它可以傳入兩個參數，而參數的類型皆為正整數。
2	回傳兩個參數相加的結果。
3	呼叫函式「Sum」，並且給予參數值10與20，並且輸出回傳的結果。

7-5參數

- 參數就像函式的入口，也可以當做只屬於函式本身能使用的變數（實際上它算是傳入資料的替身），外部的程式要把資料放進去時，都必須要透過參數來做溝通。



7-5參數

- 函式傳入的參數類型可以自己定義，參數的類型可以包含數字、字串甚至是陣列，定義參數的類型就像是定義函式入口的形狀，資料型態不符合的值無法傳到函式中處理。
- 變數傳進函式之後，本身並不會被修改，但是陣列會。

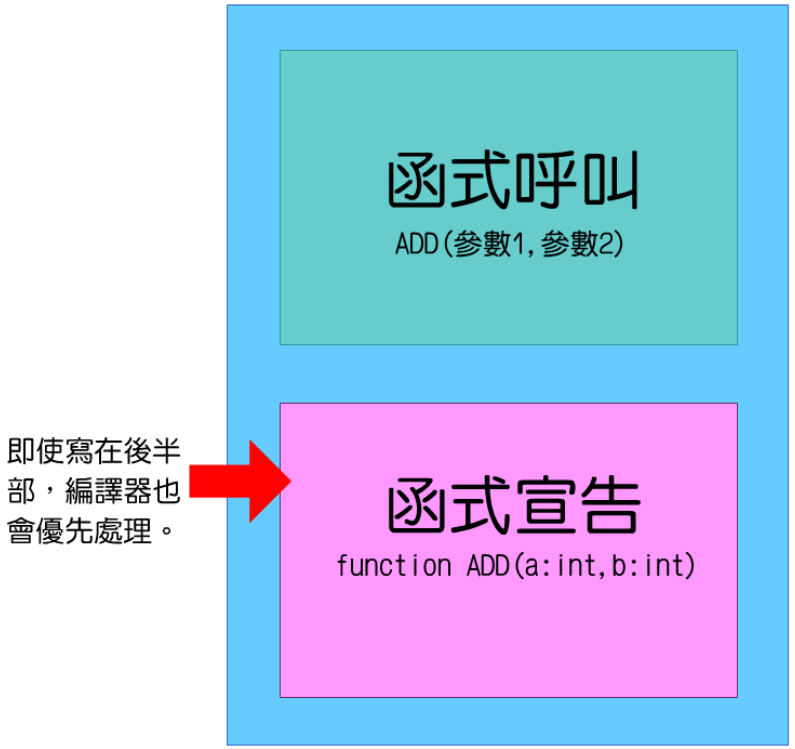
7-6回傳值「return」

- 函式可以區分為需要回傳值跟不需要回傳值，像第二章中提到的操作按鈕時的函式就屬於不需要回傳值的函式。
- 回傳的指令如下：

  return 回傳的資料（變數，或是變數的運算式）;
- 函式中執行到「return」指令，後面的程式碼都不會執行了。

7-7函式在程式碼裡的順序

- 函式對電腦的編譯器而言屬於優先處理的區塊，因此只要我們建立成函式的程式碼電腦都會先處理，所以我們在撰寫程式時，會習慣將函式寫在程式碼的最後面（因為函式通常都是包裹好的部份，已經是某樣功能性的指令，就像「stop()」之類的指令），其它需要執行的部份我們則寫在比較前面，方便我們做檢查。



7-8變數的交替影響

- 直接寫在影格中的變數被稱為是「影格變數」，而在函式之中的變數被稱為是「區域變數」，在函式之中的「區域變數」無法在影格程式中直接使用。

程式碼：

```
1. function sub(){
2.   var n:int=1;
3. }
4. sub();
5. trace(n);
```

時間軸 | 編譯器錯誤 - 1 已報告

位置	說明	來源
場景 1, 圖層 '圖層 1', 影格 1...	1120: 存取未定義的屬性 n。	trace(n);

全部的 ActionScript 錯誤:1, 報告的錯誤:1

7-8變數的交替影響

- 如果我們是在影格上宣告一個變數，函式之中則可以直接使用它

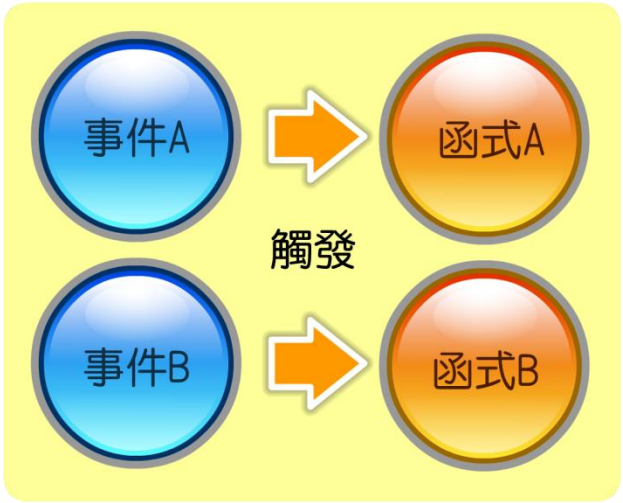
程式碼：

```
1. var n:int=1;
2. function sub(){
3.   trace(n);
4. }
5. sub();
```

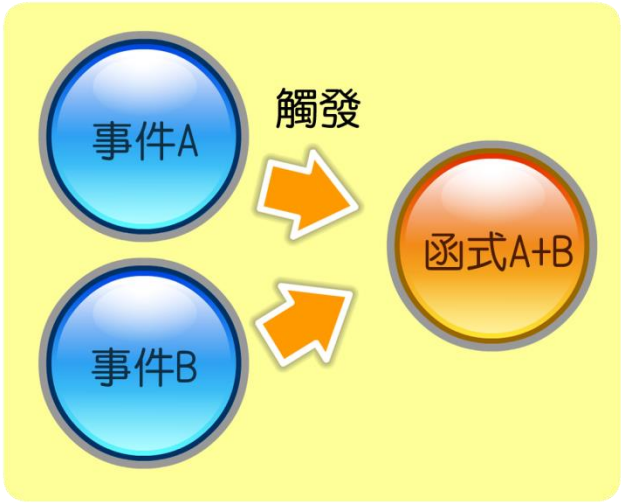
- 輸出結果：1

7-9事件與函式

- 針對事件觸發時都要有一個對應的函式來動作，而我們可以一個事件寫一個函式，也可以讓一個函式對應很多事件。



一個函式對應一個事件



一個函式對應很多事件

7-9事件與函式

- 要讓一個函式對應多種事件，最重要的條件就是要知道目前對應的元件為何，並且觸發的事件是什麼，我們可以透過以下這些事件屬性來知道這些訊息：

屬性	功用
currentTarget	代表目前事件監聽的物件。
type	被監聽事件的類型。

7-9事件與函式

程式碼：

```
1.  function btnf(event:MouseEvent) {
2.    if (event.type=="click") {
3.      trace("滑鼠按下");
4.    }
5.    if (event.type=="mouseOver") {
6.      trace("滑鼠滑過");
7.    }
8.  }
9.  btn.addEventListener(MouseEvent.CLICK,btnf);
10. btn.addEventListener(MouseEvent.MOUSE_OVER,btnf);
```

行數	程式碼的功用
1~8	宣告一個函式「btnf」，當按鈕被按下的時候，輸出字串「滑鼠按下」，當滑鼠指標滑過按鈕時，輸出字串「滑鼠滑過」。
9	按鈕「btn」加入事件監聽，當滑鼠按下時觸發事件。
10	按鈕「btn」加入事件監聽，當滑鼠滑過時觸發事件。

7-10章節案例：計算機

◎目標：

製作一台模擬現實中的計算機，具備加減乘除的運算能力。

◎程式的原理：

首先我們可以把計算機上的按鈕分成兩大類，一個是**數字(0-9)**，一個是**運算符號(+-x÷之類的)**，一開始計算機上會顯示0，讓使用者可以任意輸入數字，而當使用者按下運算符號之後，畫面上顯示的數字就會被儲存在一個暫存的變數之中，使用者再輸入新的數字之後，再按下運算符號，就會計算出答案。

計算機中比較特別的是它一次只能顯示一個數字，所以什麼時候該讓畫面上的數字可以重新輸入是整個程式中最關鍵的地方，也就是當畫面上數字為0或是使用者按下運算符號之後。

7-10章節案例：計算機

7-10章節案例：計算機

- 完成品

7-11習題

1. （　）函式的宣告，下列何者為正確？
   （Ａ）function trace(a)（Ｂ）function ball(a:int)（Ｃ）function ball(var a:int;)（Ｄ）function ball(a:int;b:int)。
2. （　）一個函式中參數最多可以有幾個？
   （Ａ）1個（Ｂ）2個（Ｃ）3個（Ｄ）無限多個。
3. （　）關於函式的敘述，下列何者為非？
   （Ａ）函式裡宣告的變數不能在函式外部使用（Ｂ）函式一定要有參數（Ｃ）函式一旦執行到「return」指令，後面的程式就不會繼續執行（Ｄ）一個函式不一定要有回傳值。
4. （　）下哪一個可以當作函式的名稱？
   （Ａ）123（Ｂ）stop（Ｃ）123abc（Ｄ）a/*/b。
5. （　）「event.type」的資料型態為？
   （Ａ）陣列（Ｂ）字串（Ｃ）整數（Ｄ）布林。

08-陣列介紹

8-1 前言

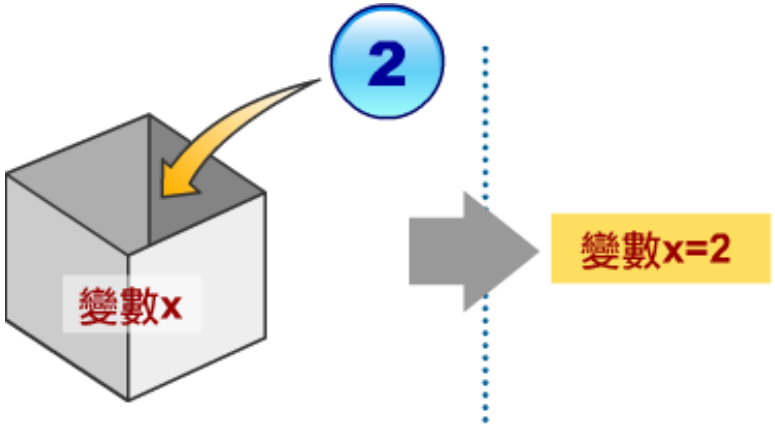
- 學會了陣列之後，我們就可以開始寫有許多資料的程式。

8-2學習目標

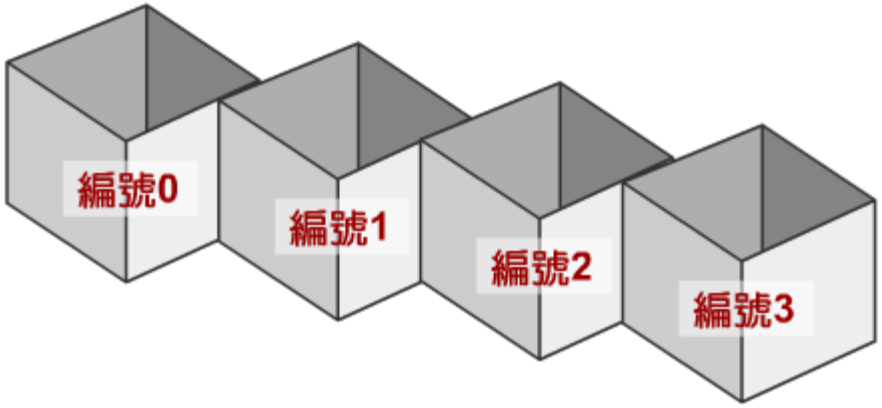
- 認識陣列。
- 利用迴圈與陣列的關連。
- 排序。

8-3 陣列的概念

- 陣列用最簡單的說法就是：帶有編號的變數。



變數就像只有一個盒子可以擺東西



陣列則是有一排櫃子可以擺東西

8-4 陣列的宣告與格式

var 陣列名稱:Array=new Array(數字);

- 括弧裡面如果寫了一個數字10，電腦就會幫我們產生編號從0到9的十個儲存空間，要注意編號是從0開始而不是從1開始，所以最後一個編號會比我們填的數字還要少1。這十個儲存空間產生的時候都是空的，而預設的資料為「undefined」這個值，表示沒有定義，我們可以用下列的寫法存入資料：

8-4 陣列的宣告與格式

- 我們也可以在宣告的時候就一併寫入資料，例如：

```
var myArray:Array=new Array(10,10,10);
```

- 電腦就會自動幫我們產生三個儲存空間，並且都存入10這筆資料

8-5利用迴圈填入資料

程式碼：
1. var myArray:Array=new Array(100); 2. for(var i:int=0;i<=99;i++){ 3. myArray[i]=i; 4. }

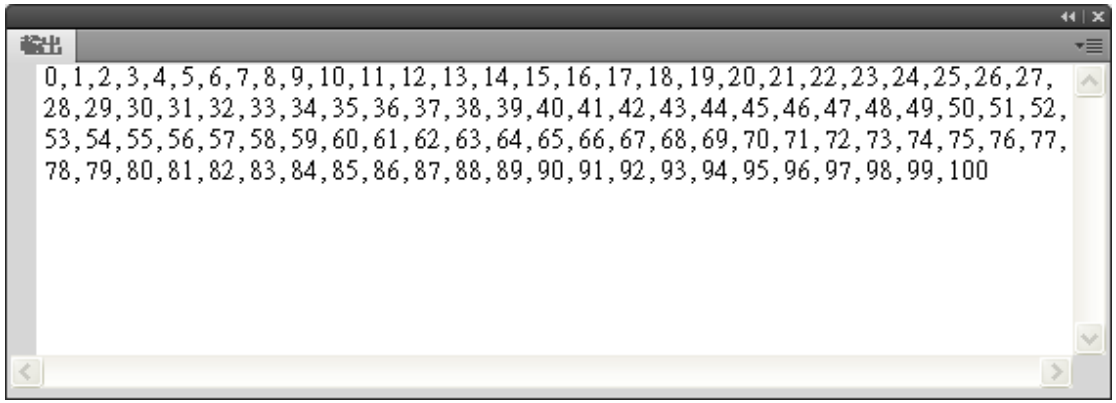
行數	程式碼的功用
1	宣告一個有一百個儲存空間的陣列。
2	設定迴圈的條件，讓這個迴圈執行一百次。
3	將變數「i」的值儲存到對應的位置（myArray[0]=0、myArray[1]=1…等）。

8-6陣列的輸出

- 我們可以使用「trace」指令，做最基本的輸出，每個資料之間會被逗號隔開，假設我們將前面已經存好資料的「myArray」輸出：

```
trace(myArray);
```

- 輸出結果：



8-6陣列的輸出

- 利用迴圈來進行資料的輸出：

```
for (var j:int=0; j<=100; j++) {
        trace("myArray["+j+"]="+myArray[j]);
}
```

- 輸出結果：

8-7陣列的資料增減

◎ push()：

```
var office:Array=new Array(“肥豬”,”企鵝”,”大熊”);
office.push(“兔子”);
trace(office);
```

- 輸出結果：
- 肥豬,企鵝,大熊,兔子

8-7陣列的資料增減

◎ **unshift()**：

```
var office:Array=new Array(“肥豬”,”企鵝”,”大熊”);
office. unshift(“兔子”);
trace(office);
```

- 輸出結果：
- 兔子,肥豬,企鵝,大熊

8-7陣列的資料增減

◎ **pop()**：

```
var office:Array=new Array(“肥豬”,”企鵝”,”大熊”);
trace(office.pop());
trace(office);
```

- 輸出結果：
- 大熊
- 肥豬,企鵝

8-7陣列的資料增減

◎ **shift()**：

```
var office:Array=new Array(“肥豬”,”企鵝”,”大熊”);
trace(office.shift());
trace(office);
```

- 輸出結果：
- 肥豬
- 企鵝,大熊

8-8 排序「sort()」

- 使用陣列通常是為了處理為數眾多的資料量，因此如何將資料整理好是程式設計師的重要課題。
- 我們分為字串跟數字這兩種資料類型來處理，「sort()」預設是以字串來做排序，因此陣列裡的資料如果是數字，也會被當作字串。

8-8 排序「sort()」

- 字串排序（以單字的第一個字母判斷）：

```
var office:Array=new Array(“pig”,”rabbit”,”bear”);
office.sort();
trace(office);
```

- 輸出結果：
- bear ,pig,rabbit

8-8 排序「sort()」

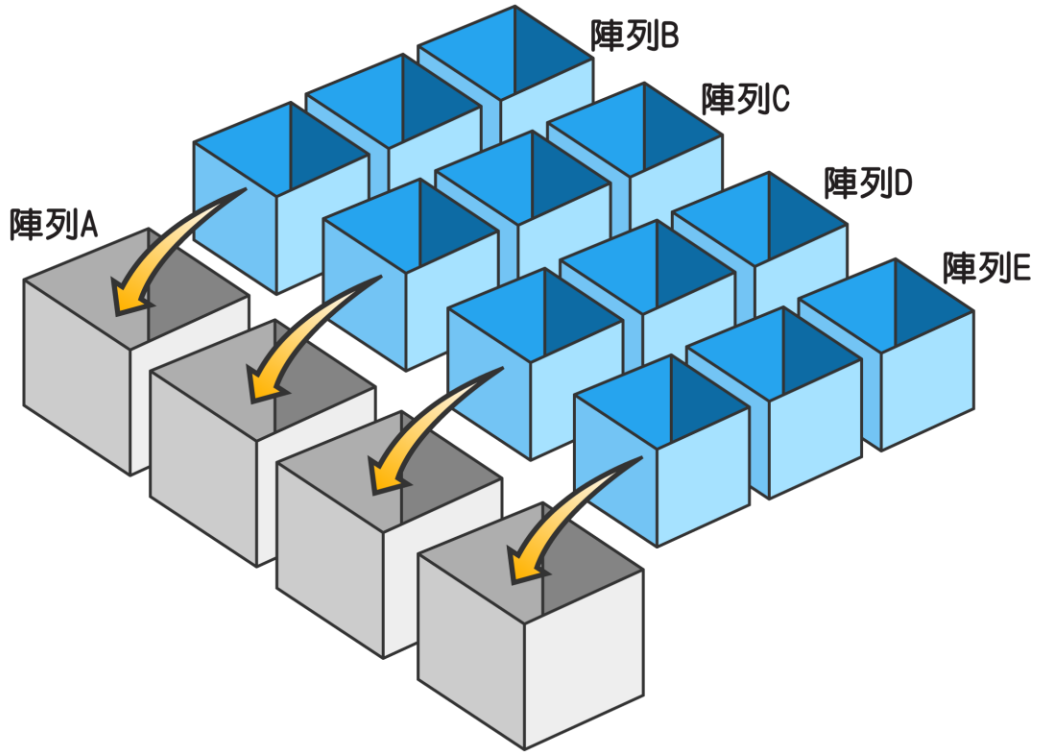
- 數字排序

```
var office:Array=new Array(10,100,50,30,1);
office.sort(Array.NUMERIC);
trace(office);
```

- 輸出結果：
- 1,10,30,50,100

8-9 二維陣列

- 二維陣列簡單的說，就是每個陣列的欄位，再存放一個新的陣列。



8-9 二維陣列

- 一維陣列儲存的資料比較像一條線，而二維陣列則把資料擴張成一個面。

程式碼：
1. var myArray:Array=new Array(2); 2. myArray[0]= new Array(1,2,3); 3. myArray[1]= new Array(4,5,6); 4. trace(myArray[0][0]);

行數	程式碼的功用
1	宣告一個有兩個儲存空間的陣列。
2	將陣列的第一個儲存空間中放入一個新的陣列。
3	將陣列的第二個儲存空間中放入一個新的陣列。
4	輸出二維陣列中第一筆資料，需要比一維陣列多一個索引參數。

8-9 二維陣列

- 二維陣列其實就像平面座標，可以參考下列這張表了解陣列中的每個值。

myArray[0][0]	myArray[0][1]	myArray[0][2]
1	2	3
myArray[1][0]	myArray[1][1]	myArray[1][2]
4	5	6

8-10章節案例：平均值計算機

◎目標：

一個可以讓使用者無限輸入數字，並且計算出平均值的簡單計算機。

◎程式原理：

- 利用陣列的方法push，將資料新增到陣列之中，而我們將陣列裡的數字全部加起來之後，再除以陣列內資料的個數（屬性length），我們就可以得到平均數。

8-10章節案例：平均值計算機

- 完成檔

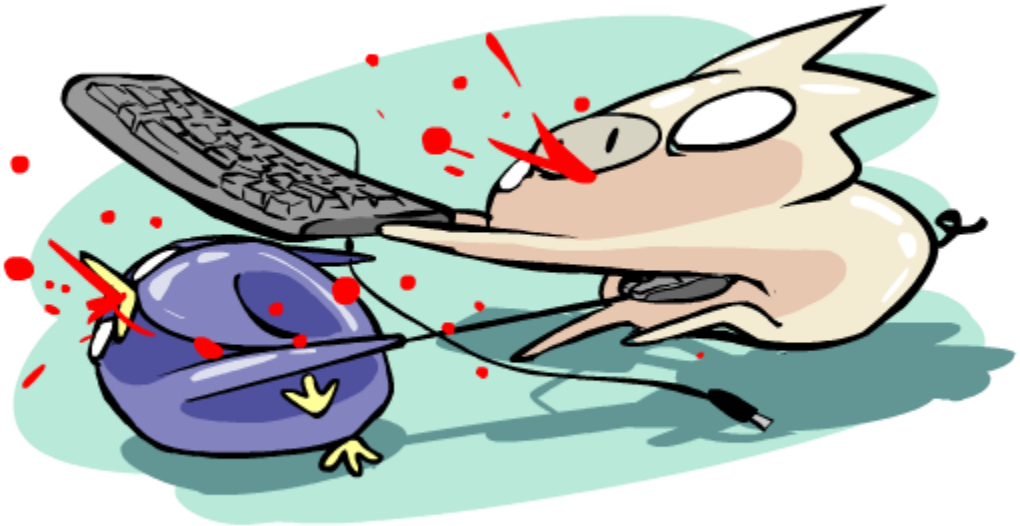
8-11習題

1. （　）「**var myArray:Array=new Array(1,2,3,4);**」，「**myArray[2]**」的值為？
（Ａ）1（Ｂ）2（Ｃ）3（Ｄ）4。
2. （　）承上題，「**myArray.pop();**」回傳的值為？
（Ａ）1（Ｂ）2（Ｃ）3（Ｄ）４。
3. （　）承上題，「**myArray.shift();**」回傳的值為？
（Ａ）1（Ｂ）2（Ｃ）3（Ｄ）４。
4. （　）使用「**sort**」指令時，陣列預設是以什麼類型來排序？
（Ａ）字串（Ｂ）數字（Ｃ）布林（Ｄ）以上皆是。
5. （　）「**var myArray:Array=new Array(10,8,100,52); myArray.sort();**」，「**myArray**」經過排序之後的結果為何？
（Ａ）10,100,8,52（Ｂ）10,100,52,8（Ｃ）8,10,52,100（Ｄ）100,10,52,8。

09-鍵盘控制

9-1 前言

- 對鍵盤的控制，AS3改成事件監聽的概念，就跟按鈕概念一樣，我們只要建立事件監聽來監聽鍵盤的一舉一動，就可以反映在程式上。



9-2 學習大綱

- 認識Flash中的鍵盤控制。
- 定時啟動的程式「setInterval」。
- 碰撞偵測「hitTestObject」。

9-3 鍵盤控制

- 我們通常讓場景（stage）加入事件監聽，因為我們不需要特別指定是哪個物件針對鍵盤的指令動作。

```
function 函式名稱(event:KeyboardEvent){
    需要執行的動作
}
stage.addEventListener(監聽事件,函式名稱);
```

9-4 鍵盤的事件

- 鍵盤的事件很單純，只有兩種：

事件	代表的意義
KeyboardEvent.KEY_DOWN	鍵盤上的按鍵被按下。
KeyboardEvent.KEY_UP	鍵盤上的按鍵被按下之後放開。

9-5 鍵盤對應「keyCode」

- 「keyCode」簡單來說，就是鍵盤上的每個按鍵其實都有對應的數字，大寫跟小寫的英文字是一樣的代表數字。
- 有了這些對應的數字，我們就可以針對特定的鍵盤按鈕下監聽指令，比方我希望當使用者按下「A」鍵時，觸發動作，那麼我們可以搭配判斷式，去判斷當鍵盤事件被啟動時，「keyCode」是否符合我們所監聽的項目（也就是「A」鍵，對應數字是65）

9-5 鍵盤對應「keyCode」

程式碼：

```
1.  function Keyreader(event:KeyboardEvent){
2.    if(event.keyCode==65){
3.      trace("執行動作" );
4.    }
5.  }
6.  stage.addEventListener(KeyboardEvent.KEY_DOWN, Keyreader);
```

行數	程式碼的功用
1	宣告一個函式「Keyreader」，指定為鍵盤事件（event:KeyboardEvent）。
2~4	如果鍵盤按下的鍵為「A」，則輸出字串「執行動作」。
6	場景加入對鍵盤事件的監聽，當鍵盤被按下放開之後啟動函式「Keyreader」中指定的動作。

9-6 特殊按鈕：「Keyboard」類別

- 某些特殊按鍵可以透過偵測「Keyboard」類別中的常數來判斷，好處就是程式碼比較容易閱讀，「Keyboard」類別也是不用宣告實體就可以使用的類別。

程式碼：

```
1.  function Keyreader(event:KeyboardEvent){
2.    if(event.keyCode== Keyboard.ENTER){
3.      trace("執行動作" );
4.    }
5.  }
6.  stage.addEventListener(KeyboardEvent.KEY_DOWN, Keyreader);
```

行數	程式碼的功用
1	宣告一個函式「Keyreader」，指定為鍵盤事件（event:KeyboardEvent）。
2~4	如果鍵盤按下的鍵為「Enter」，則輸出字串「執行動作」。
6	場景加入對鍵盤事件的監聽，當鍵盤被按下放開之後啟動函式「Keyreader」中指定的動作。

9-7 定時啟動的程式「setInterval」

- 我們在撰寫遊戲的時候，常常會需要程式持續做某些動作。
- 「setInterval」函式會以毫秒（千分之一秒）為單位來執行指定的動作。

```
function 函式名稱(){
    執行動作
}
setInterval(函式名稱,時間長度(單位是毫秒));
```

9-7 定時啟動的程式「setInterval」

- 連續輸出字串:

程式碼：
1. function output(){ 2. trace(“Hello”); 3. } 4. setInterval(output,1000);

行數	程式碼的功用
1	宣告一個函式「output」。
2	輸出字串「Hello」。
3	每隔一秒執行一次函式「output」，因此每隔一秒會輸出字串「Hello」。

9-7 定時啟動的程式「setInterval」

- 停止「setInterval」函式:

程式碼：

```
1.  var n:int=1;
2.  function output(){
3.    if(n<=10){
4.     trace(“Hello”);
5.     n=n+1;
6.    }else{
7.     clearInterval(intervalId);
8.    }
9.  }
10. var intervalId:uint=setInterval(output,1000);
```

行數	程式碼的功用
1	宣告一個變數「n」當作起始值。
2	宣告一個函式「output」。
3~5	如果變數「n」值小等於10，那麼就輸出字串「Hello」，並且把變數「n」的值加1。
6~8	如果變數「n」值大於10，則停止「setInterval」繼續執行。
10	變數「intervalId」儲存「setInterval」的身份認證以供「clearInterval」函式停止「setInterval」。 要注意的是「setInterval」回傳的參數的資料型態是「unit」，當程式執行10次之後，「n」就會因為超過10，而終止輸出字串。

9-8 碰撞偵測「hitTestObject」

- 「hitTestObject」的功用就是判斷兩個元件是否有碰觸在一起，如果有的話就會回傳「true」，沒有的話，就會回傳「false」。
- 碰撞偵測在製作遊戲方面非常的實用，可以製作簡單的動作遊戲。

9-8 碰撞偵測「hitTestObject」

- 假設場景上有兩個元件「mcA」與「mcB」，可以利用以下的程式來判斷這兩個元件是否有碰觸在一起：

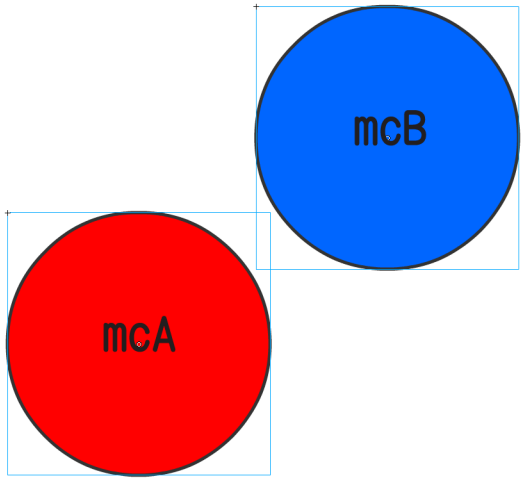
程式碼：

```
1. if(mcA.hitTestObject(mcB)){
2.    trace(“有碰在一起”);
3. }else{
4.    trace(“沒有碰到”);
5. }
```

行數	程式碼的功用
1 ~2	如果「mcA」與「mcB」有碰觸在一起，輸出字串「有碰在一起」。
3~5	如果「mcA」與「mcB」沒有碰觸在一起，輸出字串「沒有碰到」。

9-8 碰撞偵測「hitTestObject」

※要注意「hitTestObject」偵測的範圍是元件的方形邊框，而不是元件的色塊形狀



雖然「mcA」與「mcB」的圓球圖形沒有相碰，但是它們構成元件的邊框已經接觸在一起了，這時「hitTestObject」會回傳「true」。

9-9章節案例：相撲遊戲

◎目標：

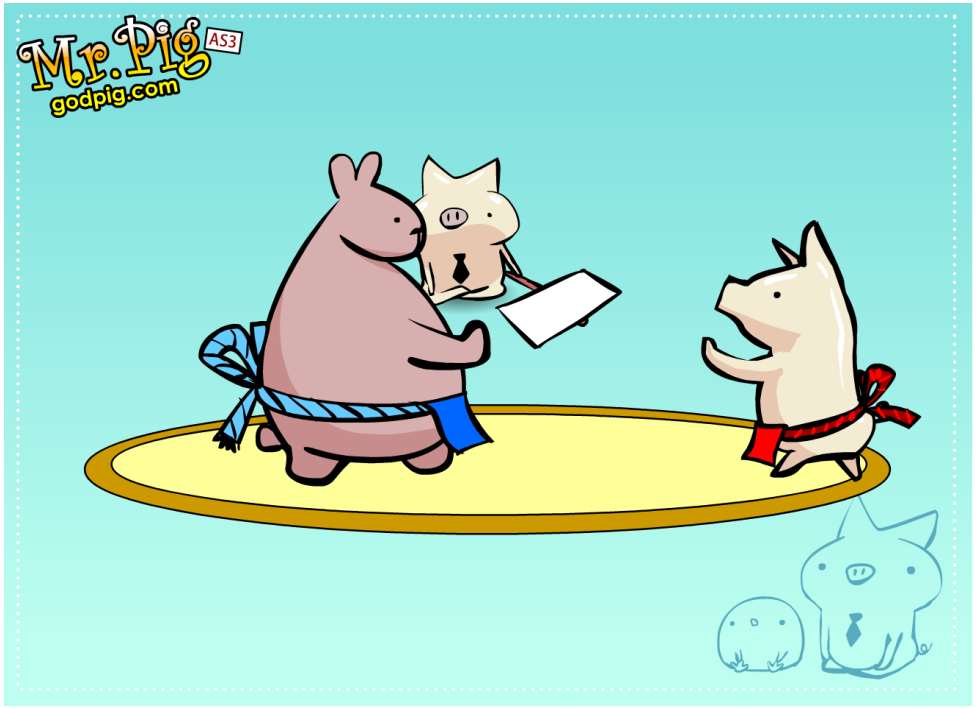
讓玩家可以透過鍵盤操作肥豬與電腦操作的大熊互相堆擠玩相撲。

◎程式的原理：

相撲遊戲的規則是在一個圓圈中，讓兩個選手互相推擠，誰先被推出圓圈就算失敗者，我們以兩個元件當做選手，玩家操作的選手，每當鍵盤按下放開，就會向左邊移動，而電腦操作的選手，則以固定的速率向右邊移動，一旦兩個元件接觸，就會讓對方以反方向移動，只要玩家按鍵盤的速度大於電腦的移動速率，就可以把電腦的選手往右邊推，直到推出我們設定的範圍，就算勝利，反之就算失敗。

9-9章節案例：相撲遊戲

- 完成品



9-10 習題

1. （　）事件「**KeyboardEvent.KEY_UP**」代表的是？
   （Ａ）鍵盤尚未被按下（Ｂ）鍵盤按下中（Ｃ）鍵盤快速按兩下（Ｄ）鍵盤按下之後放開。
2. （　）字母「**A**」的「**keyCode**」為？
   （Ａ）64（Ｂ）65（Ｃ）66（Ｄ）67。
3. （　）「**setInterval**」函式中所使用的時間單位為？
   （Ａ）秒（Ｂ）1/100秒（Ｃ）1/1000秒（Ｄ）1/10秒。
4. （　）「**setInterval**」回傳的參數的資料型態是？
   （Ａ）int（Ｂ）String（Ｃ）Number（Ｄ）uint。
5. （　）「**hitTestObject**」的用途為？
   （Ａ）判斷兩個元件是否有接觸（Ｂ）判斷兩個元件的大小（Ｃ）判斷兩個元件的距離（Ｄ）判斷兩個元件的階層。

10-聲音控制介紹

10-1 前言

- 聲音在製作Flash的時候是一個不可缺少的要素，製作專案的時候，雖然已經做完程式跟動畫的部份，還是會覺得成品很陽春，直到放上音效，整個專案作品就會變得完全不一樣。

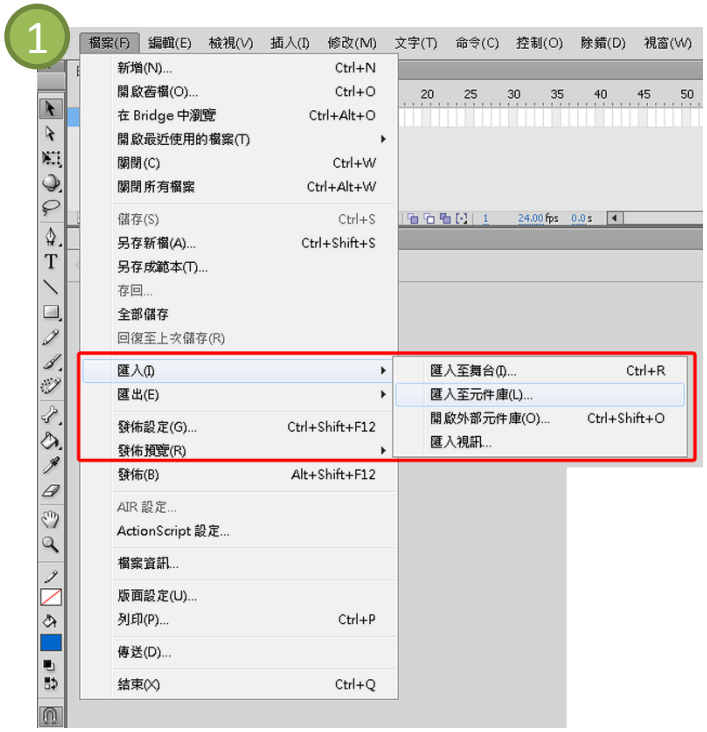
10-2 學習目標

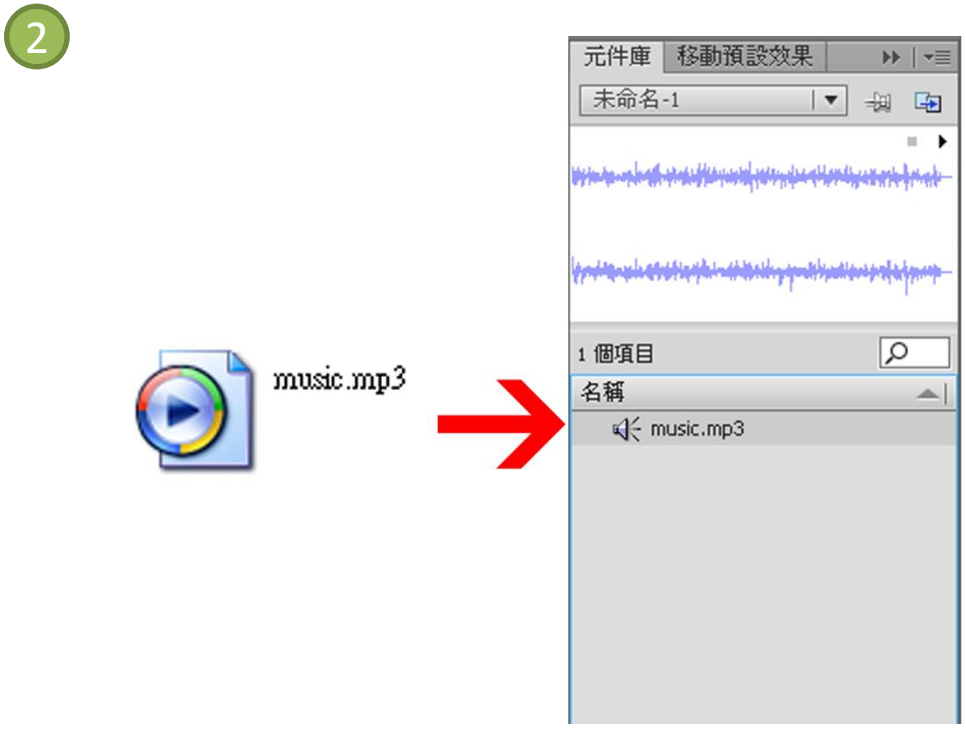
- 在Flash中控制音樂。
- 認識聲音物件「sound」。
- 組件應用。

10-3將音樂匯入Flash

- 要把音樂匯入Flash檔案中有兩種方法：

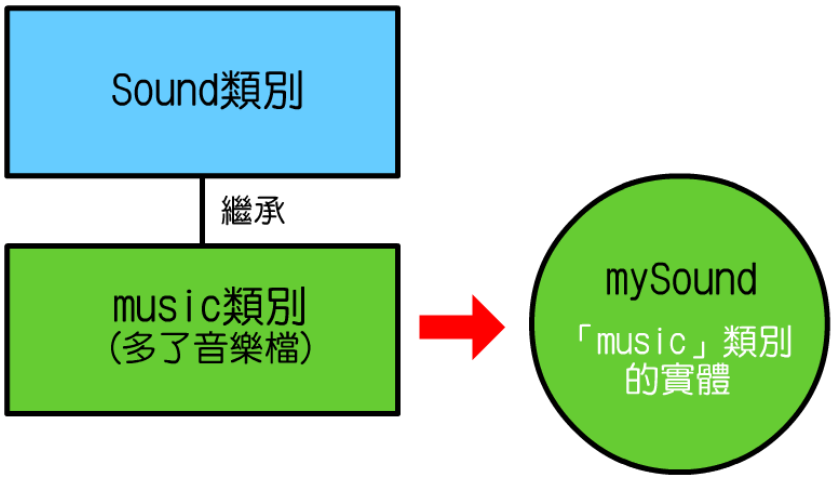
1. 使用匯入指令：
2. 直接將音樂檔，拖曳到元件庫中。





10-4 聲音物件 -「Sound」類別

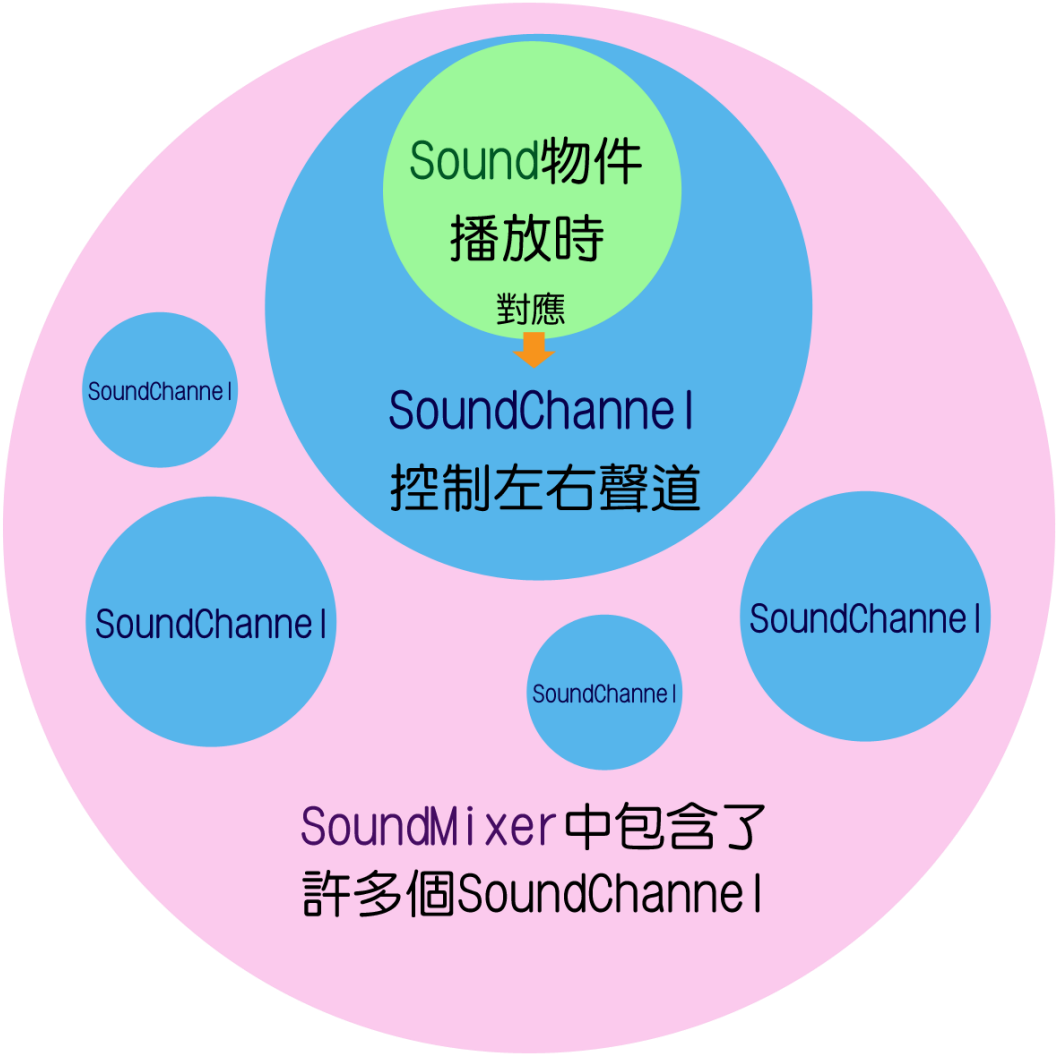
- 所有的音樂物件都是繼承至「Sound」類別，在使用之前必須要先宣告，但是如果我們是要使用在元件庫裡的聲音檔時，我們不會宣告為「Sound」類別，而是會宣告成我們自己定義的新類別。



10-4 聲音物件 -「Sound」類別

- 「Sound」類別中提供的方法只有播放而沒有停止。
- 聲音的本體是「Sound」類別，但是當它開始播放之後，程式就會產生一個「SoundChannel」物件，這個物件是用來控制聲音的播放過程，包括左右聲道的音量，以及聲音播放的停止，而每個「Sound」物件在播放時都有一個「SoundChannel」物件對應，這些多個「SoundChannel」合起來就會通過「SoundMixer」物件來做控制，也就是說「SoundChannel」物件用來

10-4 聲音物件 -「Sound」類別



10-5 聲音播放物件「SoundChannel」類別

- 要使用SoundChannel類別之前必須先宣告它，宣告方式如下：

  var 物件名稱: SoundChannel=new SoundChannel();

10-5 聲音播放物件「SoundChannel」類別

程式碼：

```
1.  var mySound:music=new music();
2.  var SdCh: SoundChannel=new SoundChannel();
3.  SdCh= mySound.play();
4.
5.  btn.addEventListener(MouseEvent.CLICK,btnf);
6.  function btnf (event:MouseEvent) {
7.          SdCh.stop();
8.  }
```

行數	程式碼的功用
1	宣告「mySound」為「music」類別的實體，而這個類別除了代表我們音樂檔案，「music」類別繼承了「Sound」類別中所有的屬性與方法。
2	宣告「SdCh」為「SoundChannel」類別的實體。
3	將「SdCh」指定為對應「mySound」物件的播放（每個「Sound」物件在播放時都有一個「SoundChannel」物件對應）。
5	將按鈕「btn」加入事件監聽，只要使用者點選就觸發動作，而監聽的事件是滑鼠點選事件，要執行的動作函式是「btnf」。
6~8	當有人呼叫函式「btnf」時，停止聲音的播放。

10-6 聲音播放設定「SoundTransform」

- 「SoundTransform」類別在宣告的時候就可以傳入兩個參數，第一個是控制音量的參數（從0到1之間的數值），0帶表靜音，1代表最大聲，第二個參數是左右相位值（從-1到1之間的數值），用來控制喇叭的左右聲道播放，-1代表聲音完全使用左聲道播放，1代表完全使用右聲道，0則代表聲音的播放沒有任何偏移。

10-6 聲音播放設定「SoundTransform」

程式碼：

```
1.  var mySound:music=new music();
2.  var SdCh: SoundChannel=new SoundChannel();
3.  SdCh= mySound.play();
4.
5.  btn.addEventListener(MouseEvent.CLICK,btnf);
6.  function btnf (event:MouseEvent) {
7.    SdCh.soundTransform=new SoundTransform(0.5);
8.  }
```

行數	程式碼的功用
1	宣告「mySound」為「music」類別的實體，而這個類別除了代表我們音樂檔案，「music」類別繼承了「Sound」類別中所有的屬性與方法。
2	宣告SdCh為SoundChannel類別的實體。
3	將SdCh指定為對應mySound物件的播放（每個「Sound」物件在播放時都有一個「SoundChannel」物件對應）。
5	將按鈕「btn」加入事件監聽，只要使用者點選就觸發動作，而監聽的事件是滑鼠點選事件，要執行的動作函式是「btnf」。
6~8	當有人呼叫函式「btnf」時，將音量設定為一半。

10-7 所有聲音的總和「SoundMixer」

- 「SoundMixer」類別在使用前不需要先宣告，它代表的是整個SWF檔中音源。
- 我們一樣可以透過「SoundTransform」類別先做好設定再回存到「SoundMixer」類別「soundTransform」屬性。
- 我們可以利用下列語法將所有聲音的音量大小設定為一半：

```
SoundMixer.soundTransform=new SoundTransform(0.5);
```

除了控制音量之外，我們也可以透過下列語法讓檔案中所有的音效停止播放：

```
SoundMixer.stopAll();
```

10-8 載入外部音檔

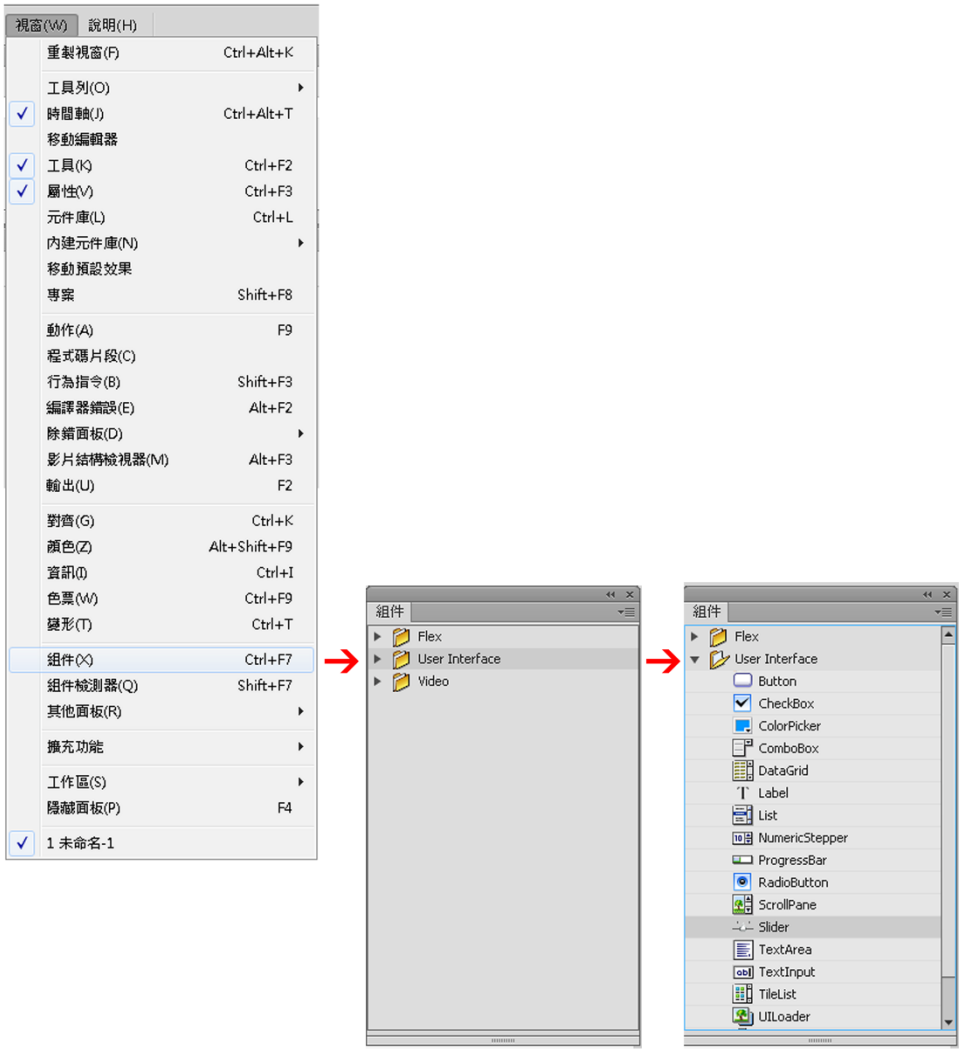
- 透過外部連結的方式載入聲音檔，主要支援的音樂格式是MP3。

程式碼：
1. var SoundURL:URLRequest=new URLRequest(“music.mp); 2. var mySound:Sound=new Sound(SoundURL);

行數	程式碼的功用
1	宣告「SoundURL」為新建立的「URLRequest」物件，而這個物件代表的外部連結路徑為「music.mp3」（因為在同一個資料夾底下，我們可以使用相對路徑）。
2	宣告「mySound」為一個新的「Sound」物件，並且載入「SoundURL」中指定的聲音檔。

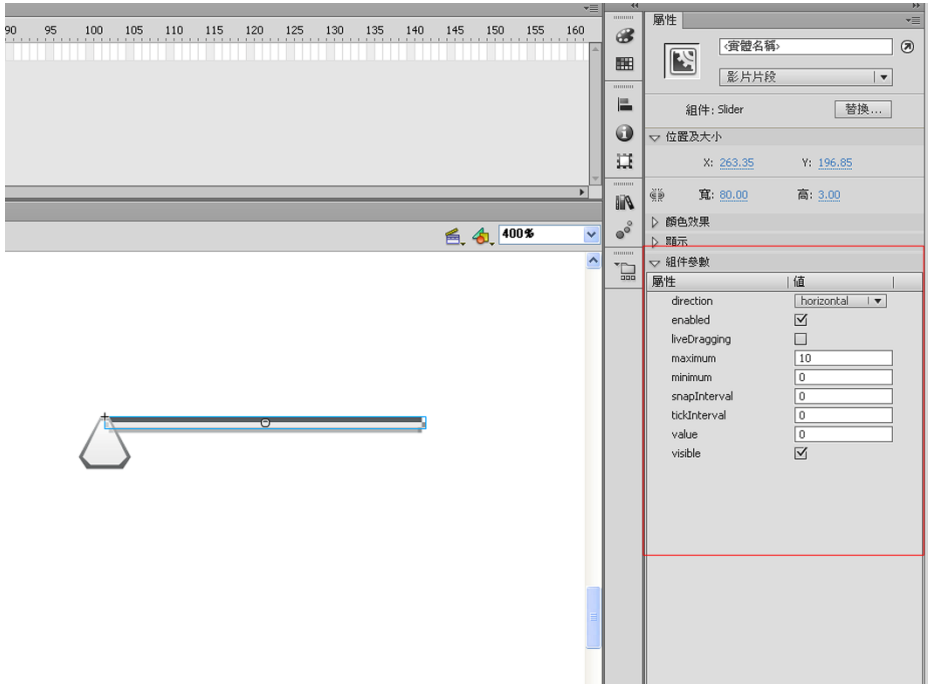
10-9 組件

- 所謂的組件其實就是由官方建立的元件，已經內建在Flash程式之中，只要從「視窗」→「組件」就可以叫出組件面版。



10-9 組件

- 在CS6之中，組件中的屬性我們可以透過屬性面版做設定，點選該組件，它的參數就會顯示在屬性面板中的組件參數上。



10-10 滑動軸「Slider」組件

- 「Slider」組件就是常見的音樂播放器上都有的滑動軸，可以讓使用者自行拖曳到他想要聽的部份。
- 要用程式來控制「Slider」組件前，我們必須要先執行匯入的指令：

```
import fl.events.SliderEvent;
```

10-10 滑動軸「Slider」組件

- 首先將組件取名為「SLD」。
- 「Slider」組件最重要的屬性就是它拖曳到最右邊時的最大值是多少，以及拖曳到最左邊的最小值:

  SLD.maximum=100;

  SLD.minimum=0;

10-10 滑動軸「Slider」組件

程式碼：

```
1. import fl.events.SliderEvent;
2. SLD.maximum=100;
3. SLD.minimum=0;
4. SLD.addEventListener(SliderEvent.CHANGE,SLDf);
5. function SLDf(event:Event) {
6.   trace(SLD.value);
7. }
```

行數	程式碼的功用
1	匯入「Slider」組件相關的檔案。
2	設定組件「SLD」的最大值為100。
3	設定組件「SLD」的最小值為0。
4	將組件「SLD」加入事件監聽，當使用者拖曳「SLD」的橫桿時，執行動作函式「SLDf」。
5~7	輸出組件「SLD」被拖曳時改變的值。

10-11章節案例：音樂播放器

◎目標：

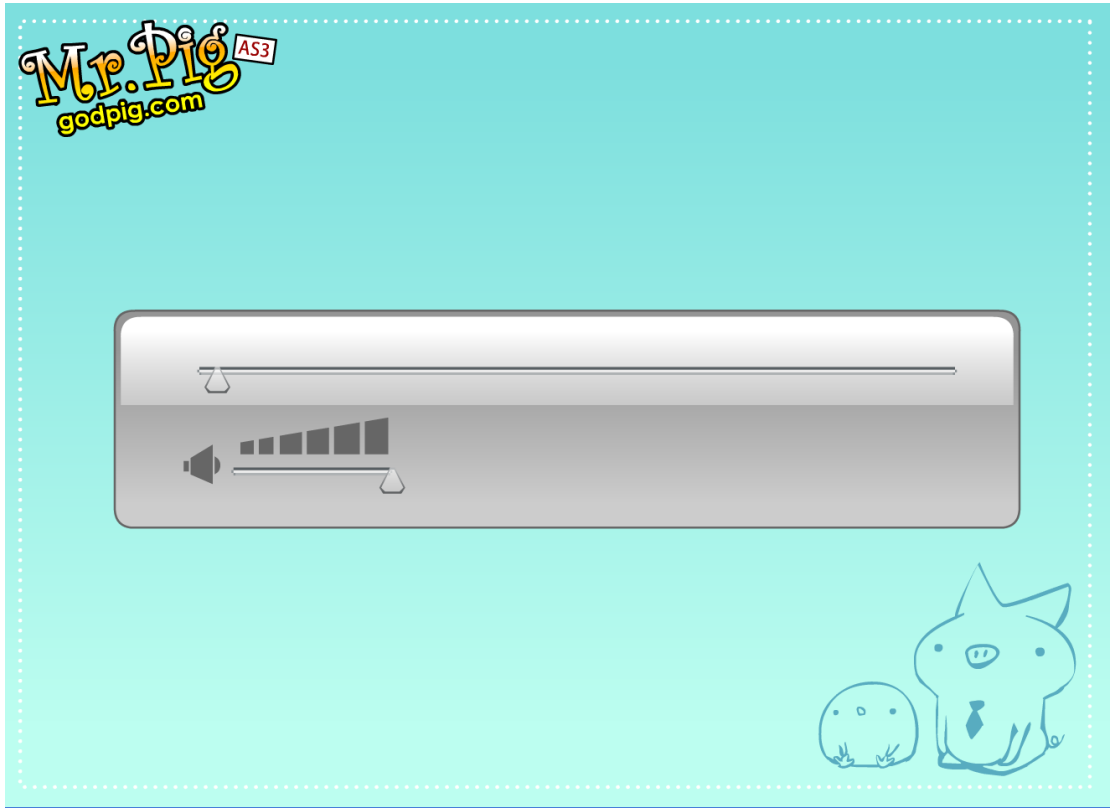
利用**滑動軸組件**製作音樂播放器，具備聲音拖曳播放的時間軸與音量大小聲的控制。

◎程式的原理：

Slider組件提供了滑動軸完整的功能，所以我們只需要將使用者拖曳時的數字去做對應即可製作出音樂播放器，其中在音樂播放的同時，時間軸會移動是因為我們使用了**ENTER_FRAME**的事件去監聽目前音樂的播放進度，再把進度轉換成滑動軸位移的長度。**(ENTER_FRAME事件只要一啟動就會持續觸發，很類似setInterval)**

10-11章節案例：音樂播放器

- 完成品



10-12習題

- （　）**Flash**支援下列哪一種音樂格式？
（Ａ）midi（Ｂ）wma（Ｃ）mp3（Ｄ）m4a。
- （　）所有匯進**Flash**的音檔，都是繼承至哪一個類別？
（Ａ）MovieClip（Ｂ）Sprite（Ｃ）Math（Ｄ）Sound。
- （　）當我們要控制整體**Flash**的音量，可以使用下列哪個類別？
（Ａ）SoundMixer（Ｂ）SoundChannel（Ｃ）Sound（Ｄ）以上皆可。
- （　）下列哪一個事件代表載入完畢？
（Ａ）Event.CANGE（Ｂ）Event.COMPLETE（Ｃ）Event.CONNECT（Ｄ）Event.ENTER_FRAME。
- （　）「**Slider**」組件屬性中，設定最大值的屬性為？
（Ａ）minimum（Ｂ）enabled（Ｃ）value（Ｄ）maximum。