SpaceShooter2D 单机版实作范例

- 1-1 建立游戏项目基础架构
- 1-2 玩家角色
- 1-3 敌人系统
- 1-4 碰撞处理
- 1-5 游戏机制与 GUI 设计
- 1-6 细部调整

大风起兮云飞扬 威加海内兮归故乡

安得猛士兮守四方

《大风歌》

本文将以 Unit3D 的 3D 场景制作一个 2D 太空射击游戏,透过实作 让读者更能了解如何以 Unity3D 开发游戏。除了与美术有关的模型与 素材外,这个游戏项目将会从无到有,一步一步带领读者实际操作,完 成太空射击游戏。

接着我们就开始进行,让大家体会开发游戏的乐趣吧!

1-1 建立游戏项目基础架构

- 1-2 玩家角色
- 1-3 敌人系统

1-4 碰撞处理

完整教程请至 http://developer.arcalet.com 进行下载。

制造爆炸效果

为了让游戏看起来更加生动,当炮弹击中陨石,或陨石击中太空战机,撞击的地方应该要出现爆 炸的效果,在为游戏加上这个特效之前,我们先来制作一个产生爆炸效果的 Prefab。

首先在场景中建立一个 Particle System,	同时将名称改为 Explosion:

GameObject Component Terr	ain Window H	
Create Empty	Ctrl+Shift+N	
Create Other	•	Particle System
Center On Children		Camera
Make Parent		GUI Text
Clear Parent		GUI Texture
Apply Changes To Prefab		3D Text
Break Prefab Instance		00 102
Move To View	Ctrl+Alt+F	Directional Light
Align With View	Ctrl+Shift+F	Point Light
Align View to Selected		Spotlight
		spotlight
		Area Light
		Cube

接着按照以下步骤修改特效属性:

(1) 请设定基本属性:

🖲 👸 🏾 Particle System	Open Edito	2] \$, r
Explosion		+
Duration	0.50	
Looping		
Prewarm		
Start Delay	0	
Start Lifetime	0.5	•
Start Speed	2	•
Start Size	0.7	•
Start Rotation	0	•
Start Color		•
Gravity Modifier	0	
Inherit Velocity	0	
Simulation Space	Local	•
Play On Awake	×.	
Max Particles	500	

(2) 设定 Start Color 为 Gradient,并依下图调整分段颜色与透明度



(3) 设定 Emission 属性

✓ Emission			
Rate	100		•
	Time		•
Bursts	Time	Particles	
		0	

(4) 依下图设定 Shape 属性

🗸 Shape		
Shape	Sphere	•
Radius	1	
Emit from Shell		
Random Direction		

(5) 设定 Color Over Lifetime 为 Gradient,并调整分段颜色与透明度

\checkmark	Gradient
	Random Between Two Gradients



(6) 设定 Size Over Lifetime 属性 (请记得展开下方的「Particle System Curves」)



(7) 设定 Renderer 的 Material,选择 Unity3D 所提供标准粒子系统的「Fire3」Asset

🗸 Renderer		
Render Mode	Billboard	•
Material	Garage G Garage Garage G Garage Garage Ga	0
Sort Mode	None	•
Sorting Fudge	0	
Cast Shadows	×	
Receive Shadows	×	
Max Particle Size	0.5	

如果「Fire3」Asset 没有出现在选单里面,请点选主选单→「Assets」→「Import Package」→「Particles」, 然后把它们全部汇入项目中。

Create	+	Custom Package	
Show in Explorer			
Open		Character Controller	
Delete		Glass Refraction (Pro Only)	
Import New Asset		Image Effects (Pro Only)	
Import Package	۲	Light Cookies	
Export Package		Light Flares	
Find References In Scene		Particles	
Select Dependencies		Physic Materials	
Refresh	Ctrl+R	Projectors	把 P (1) 素特尔)因素常义也
Reimport		Scripts	把 Particles 系材汇入到专条当中。
Reimport All		Skyboxes	
Sync MonoDevelop Project		Standard Assets (Mobile)	
		Terrain Assets	
		Toon Shading	
		Tree Creator	
		Water (Basic)	
		Water (Pro Oph)	

完成以上步骤后,爆炸效果就接近完成了。不过我们还要设定爆炸效果在播放结束能够自动从场 景中删除。

在 Unity3D 3.5 版以前,多数的开发者建立粒子系统时都会使用「Particle Animator」Component 做动态特效。「Particle Animator」里有个「Autodestruct」属性,它可以让动态生成的粒子系统在 播放完毕之后自动删除。也就是说,旧版粒子系统有自动删除的功能,但是我们现在以 Unity3D 3.5 版以后的粒子系统打造的爆炸特效并没有这项功能,解决方法之一就把「Particle Animator」再加 进来,不过我们只会使用它的「Autodestruct」功能,其它功能全部都用不到。另外一个方法是使 用 particleSystem.IsAlive()来判断粒子系统是否还在播放,如果不是的话就呼叫 Destroy() 函式把自己删除,这个方法虽然需要用到脚本程序,但是只是在 Update()函式中加上一小段判断 式,但是效能远比起第一种方法好,建议开发者尽量使用这个方法。现在我们依照这个方法另外 制作脚本 Explosion.cs,然后将它附加到目前我们还在调整的 Partical System 里:

```
// Explosion.cs
using UnityEngine;
using System.Collections;
public class Explosion : MonoBehaviour {
    void Update () {
        if (!particleSystem.IsAlive()) Destroy(gameObject);
        }
}
```

完成以上步骤后,爆炸特效就完成了。接着我们要把它做成 Prefab,做法大家也应该都很熟悉了,如果不熟练,那就再快速复习一遍。

首先在 Project 窗口中建立一个新的 Prefab, 然后更名为 ExplosionPrefab, 接着把 Hierarchy 窗口中的 Explosion 拖曳到 ExplosionPrefab 里,爆炸效果 Prefab 做好了,完完成后因为场景中已经不再 需要 Explosion 了,请记得把它从场景中删除。

ExplosionPrefab 完成后,我们打算把它用在陨石被炮弹击中时,为了让游戏不要太呆板,我们现

在要另外制作一个不同的爆炸效果给太空战机被陨石击中时使用。做法是直接复制 ExplosionPrefab (快速复制可以使用快捷键 Ctrl-D),然后更名为 ExplosionPrefab2,然后再依照个 人喜慢慢的调整它的爆炸效果。



爆炸音效

爆炸视觉效果完成之后,为了让爆炸效果更有临场感,我们接着为它加上爆炸的音效。爆炸的声音文件我们已经事先准备好了,就存放在 spacebag 里,请按照以下步骤将音效加进来:

(1) 选取 ExposionPrefab

- (2) 点选主选单→「Component」→「Audio」→ 「Audio Source」,此时「Audio Source」的 属性窗口就会出现在 ExposionPrefab 里。查 看一下它的「Audio Clip」的内容应该是 「none」,表示目前并没有指定任何音效。
- (3) 点选「Audio Clip」旁的小圆钮,出现选取 Camera-Control 音效的窗口,里面会出现目前项目里面可用的声音文件,请选 取「Mortar explosions 迫击炮 3 声」这个档案。
- (4) 确认「Play On Awake」是否已勾选,如果没有,请将它勾选起来,当炮弹被生成时,音效也会马上出现。
- (5) 选取 ExposionPrefab2, 并重复上面的动作, 为 ExposionPrefab2 加上相同的声音文件。



让碰撞后产生爆炸

现在我们已经把爆炸效果的 prefab 做好了,所以要让炮弹和陨石碰撞后在画面出现爆炸效果,只要在 Projectile 脚本的 OnTriggerEnter()事件函式中呼叫 Instantiate()产生 ExplosionPrefab 的实例物体,爆炸效果就会出现在画面上。因为是陨石爆炸,所以 ExplosionPrefab 生成的位置就直接取用 Enemy 的坐标,程序代码如下:

Instantiate(ExplosionPrefab, enemy.transform.position, enemy.transform.rotation);

以下是完整的 Projectile 脚本内容:

```
//Projectile.cs
using UnityEngine;
using System.Collections;
public class Projectile : MonoBehaviour {
   public float ProjectileSpeed;
   public GameObject ExplosionPrefab;
   private Transform myTransform;
   private Enemy enemy;
            ....(略) ....
   void OnTriggerEnter(Collider otherObject) {
      if(otherObject.tag == "enemy") {
           Instantiate(ExplosionPrefab, enemy.transform.position, enemy.transform.rotation);
           enemy.SetPositionAndSpeed();
           Destroy(gameObject);
       }
    }
```

```
程序中我们定义了一个公共变量:
```

public GameObject ExplosionPrefab;,

所以执行前必须指定初值。指定初值的操作方法我们也做过很多次了,先到 Project 窗口选择 ProjectilePrefab,然后在 Inspector 窗口里找到「ProjectilePrefab (Script)」 Component 中的属性字 段「Explosion Prefab」,目前还没指定初值应该显示为「none」,然后直接以鼠标将 Project 窗口的 ExplosionPrefab 拖曳到这个属性字段里,完成指定初值的动作。



现在我们继续将陨石与太空战机的碰撞处理也加上爆炸的效果,同样也是在发生碰撞时,以当时 陨石的坐标产生爆炸。以下是修改后的 Player 脚本内容:

// Player.cs
using UnityEngine;
using System.Collections;
public class Player : MonoBehaviour {
public float PlayerSpeed;
public GameObject ProjectilePrefab;
public float shipInvisibleTime;
public GameObject ExplosionPrefab;
(略)
void OnTriggerEnter(Collider otherObject) {
if(otherObject.tag == "enemy") {
Instantiate(ExplosionPrefab, enemy, transform, position, enemy, transform, rotation);
enenv.SetPositionAndSpeed();
<pre>StartCoroutine("DestroyShip");</pre>
}
J
(田久)
("")

最后别忘了指定公共变量 ExplosionPrefab的值,稍早之前我们已经为陨石击中太空飞机制作另一个爆炸效果的 prefab,名称为 ExplosionPrefab2,现在请将它指定给公共变量 ExplosionPrefab:



玩家重生后的无敌状态

无敌状态是玩家角色失败后重生后的贴心设计,如果没有这样的设计,玩家角色失败后可能会重 生在敌人环伺的危险状态中,让玩家来不及反应。为了维持游戏平衡,无敌状态是非常重要而且 必要的设计。

设计无敌状态程序代码,首先我们在 Player 脚本中定义一个列举常数,用来代表太空战机的状态,状态总共有三种,分别是「游戏进行中」、「爆炸中」、「无敌」,程序代码写法如下:

enum State {Playing, Explosion, Invincible}

接着定义一个代表目前状态的变量,变量名称为 State,初始值为 Playing: private State state = State.Playing;

然后修改 OnTriggerEnter()内的 if 判断式,增加一个 and 条件,当 State 值是 Playing 时才进行碰撞处理,如果是其它两种状态时就不处理,不处理就等同于没有碰撞发生,也就是无敌状态了:

if (otherObject.tag == "enemy" && state == State.Playing)

至于要如何让太空战机在爆炸重生后进入数秒钟的无敌状态呢?最简单的方法就是直接以

yield return new WaitForSeconds();

将控制权交回给系统,因为爆炸重生处理是在 IEnumerator DestroyShip()函式中进行,所以不 必担心线程被占用。

另外还有一个方法,就是使用 while()循环,循环每个循环都将控制权交回给系统,同时切换太 空战机为可见与不可见,这样子还可以让太空产生闪烁的效果,提醒玩家目前太空战机处于无敌 状态:

```
while (...) {
   gameObject.renderer.enabled = !gameObject.renderer.enabled;
   yield return new WaitForSeconds(0.1f);
}
```

上面这段程序代码中, WaitForSeconds(0.1f)表示每个循环等待的时间是 0.1 秒,也就是太空战机闪烁的速度,秒数越短,闪烁得越快,反之则闪得越慢。

最后再加上一些计数器来控制 while()循环的执行次数,无敌状态的程序代码就完成了:

```
// Player.cs
using UnityEngine;
using System.Collections;
public class Player : MonoBehaviour {
           ....(略) ...
    enum State {
      Playing, Explosion, Invincible
    3
   private State state = State.Playing;
   private float blinkRate=0.1f;
   private int numberOfTimesToBlink = 10;
   private int blinkCount=0;
    void OnTriggerEnter(Collider otherObject) {
       if (otherObject.tag == "enemy" && state == State.Playing) {
           Instantiate(ExplosionPrefab, enemy.transform.position, enemy.transform.rotation);
           enemy.SetPositionAndSpeed();
           StartCoroutine("DestroyShip");
       }
   }
    IEnumerator DestroyShip() {
       state=State.Explosion;
       gameObject.renderer.enabled = false;
       yield return new WaitForSeconds(shipInvisibleTime);
       transform.position = new Vector3(0.0f, transform.position.y, transform.position.z);
       gameObject.renderer.enabled = true;
       state=State.Invincible;
       while (blinkCount < numberOfTimesToBlink) {</pre>
           gameObject.renderer.enabled = !gameObject.renderer.enabled;
           if (gameObject.renderer.enabled == true) blinkCount++;
           yield return new WaitForSeconds(blinkRate);
       blinkCount=0;
       state=State.Playing;
    }
```

1-5 游戏机制与 GUI 设计

完整教程请至 http://developer.arcalet.com 进行下载。



教程从「单机模式」改编成「Online」多人实时都有详细的教程,千万别错过。

