欢迎继续我们的学习。

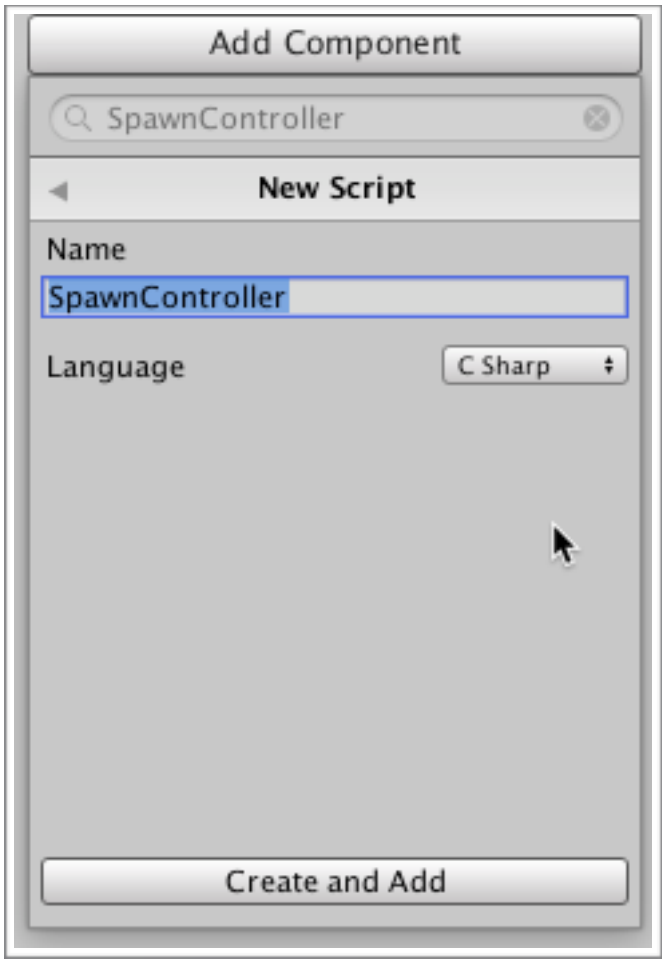
在之前的课程中我们提到过，对于游戏中的角色，最好是使用prefab预设体的方式在程序中生成，而不是直接放置在场景里面。

在这一课的内容中，我们将实现这一点。

打开Unity,在Project视图中右键单击Assets，创建一个新的文件夹，将其命名为_Prefabs。之所以加下划线，之前反复强调过，是为了将开发者自己添加的游戏资源和第三方插件中的游戏资源区分开。

从Hierarchy视图中将z@walk游戏对象拖到Project视图中的_Prefabs文件夹中，从而创建一个预设体，将其更名为zombieEnemy。然后从Hierarchy视图中删除该对象。

在Hierarchy视图中点击HItCubParent对象，选择ruined_house，在Inspector视图中点击Add Component，添加一个新的脚本，并将其命名为SpawnController。



然后在MonoDevelop中打开该脚本文件，并更改其中的代码如下：

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//1.导入UI相关的命名空间文件
using UnityEngine.UI;


public class SpawnController : MonoBehaviour {

	//2.创建到敌人的引用
	public GameObject zombie;

	//3.创建到开始游戏按钮的引用
	public Button btnStartGame;



	// Use this for initialization
	void Start () {

		//4.添加点击事件的响应
		btnStartGame.onClick.AddListener(StartInvoke);

	}

	//5.处理开始游戏

	void StartInvoke(){



	}

}
```

按照注释行的数字编号来简单解释一下：

（1）导入跟UI相关的命名空间
（2）创建了一个GameObject对象，也就是对敌人对象的引用
（3）创建了对开始游戏按钮的引用
（4）添加了对开始游戏按钮点击事件的响应机制
（5）该方法将用于处理点击事件

注意到这里我们把默认提供的Update方法删除了，因为暂时不需要用到。

接下来回到Unity编辑器，把_Prefabs文件夹中的zombieEnemy预设体拖动到ruined_house对象的Spawn Controller组件的Zombie属性处，如图所示。

接下来我们需要创建一个按钮。

在编辑器中的中间区域切换到Game视图。

然后在Hierarchy视图中右键单击Canvas，选择UI-Button，从而添加一个按钮，将其命名为btn_StartGame。

将按钮的文本内容更改为Start Game，并作出以下设置：
（1）将btn_StartGame按钮的Image组件的Source Image设置为SF Window
（2）设置btn_StartGame按钮的Rect Transform的Width和height为500，200
（3）设置文本字体为Jupiter
（4）设置字体大小为80
（5）设置文本的Color为纯白色。

在Hierarchy视图中选择HitCubeParent下面的ruined_house对象，并将Spawn Controller组件的Btn Start Game属性设置为刚刚添加的btn_StartGame按钮对象。

接下来打开SpawnController.cs，并更改其中的StartInvoke方法，修改后的代码如下：

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//导入UI相关的命名空间文件
using UnityEngine.UI;


public class SpawnController : MonoBehaviour {

	//创建到敌人的引用
	public GameObject zombie;

	//创建到开始游戏按钮的引用
	public Button btnStartGame;



	// Use this for initialization
	void Start () {

		//添加点击事件的响应
		btnStartGame.onClick.AddListener(StartInvoke);

	}


	//处理开始游戏
	void StartInvoke(){

		//1.每5秒调用一次SpawnEnemy方法
		InvokeRepeating ("SpawnEnemy", 0f, 5f);

	}

	//在场景中生成敌人
	void SpawnEnemy(){

		//2.设置敌人的生成位置
		Vector3 position = new Vector3 (Random.Range (-10f,
10f), Random.Range (-3f, 3f), Random, Range (-10f, 10f));
```

```
            //3.在特定的位置生成敌人
            Instantiate (zombie, position, Quaternion.Euler (0, 0,
0));

    }

}
```

接下来按照注释行编号简单解释一下：

1.这里使用InvokeRepeating方法来重新调用SpawnEnemey方法。
关于InvokeRepeating方法的使用，大家可以在官方文档中搜索，https://docs.unity3d.com/ScriptReference/MonoBehaviour.InvokeRepeating.html

2.这里创建了一个Vector3类型的变量 ，从而设置了敌人的生成位置。
同样的，关于Vector3，可以在官方文档中查看
https://docs.unity3d.com/ScriptReference/Vector3-ctor.html

3.使用Instantiate方法在特定的位置生成敌人。
https://docs.unity3d.com/ScriptReference/Object.Instantiate.html

注意这里的旋转角度使用了Quaternion类型，在Unity中，使用Quaternion来代表旋转角度。

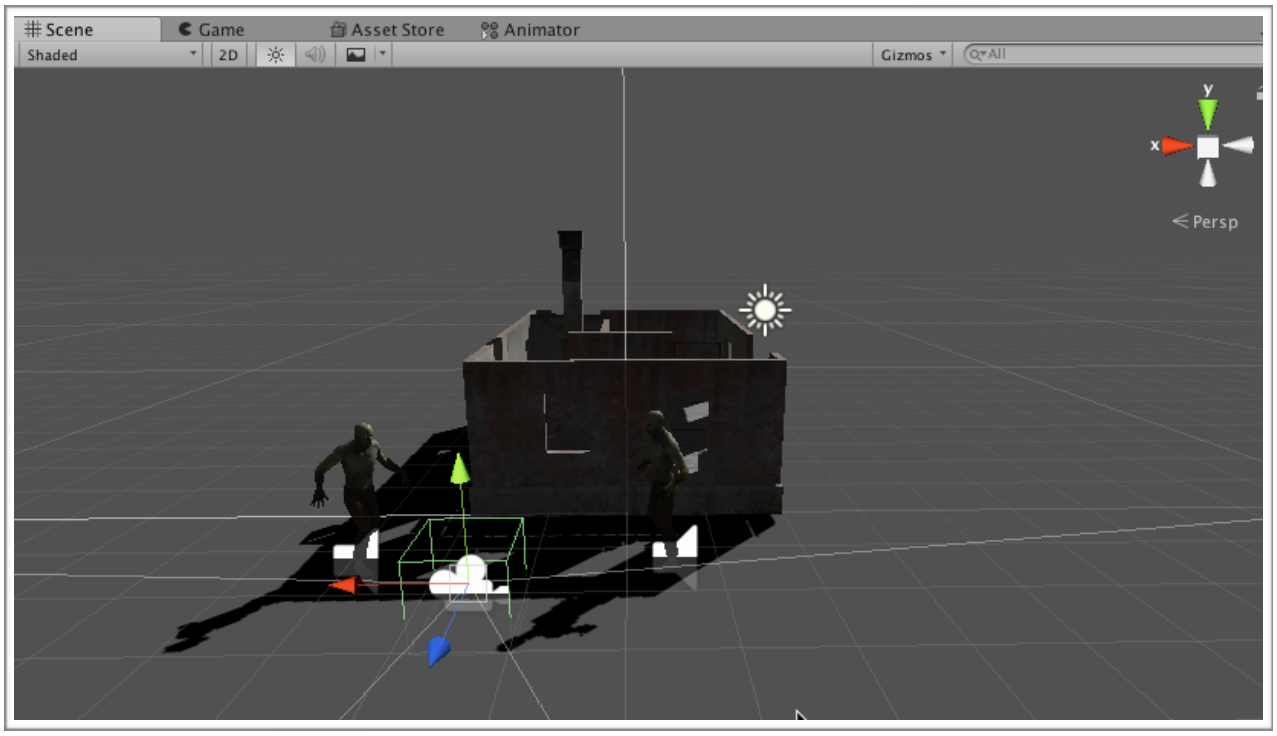
具体细节请参考官方文档：
https://docs.unity3d.com/ScriptReference/Quaternion.html

https://docs.unity3d.com/ScriptReference/Quaternion.Euler.html

现在我们可以回到Unity编辑器，点击工具栏上的Play按钮预览游戏效果。

当我们点击Start Game按钮后，就会有敌人出现在场景中，并开始之后的游戏逻辑。

如果一切顺利，我们已经成功的完成了本课所要学习的内容了~


Scene
Game
Asset Store
Animator
Shaded
2D
Gizmos
All
y
x
Persp