

SCRIPTING EN UNITY

CLASES

Ramón Mollá

rmolla at dsic.upv.es - ext. 73549

Grupo de Informática Gráfica

Departamento de Sistemas Informáticos y Computación

Índice

Programa básico

Clases en C#

C#

Clases en Unity (I). Esquema básico

MonoBehaviour, clase padre de cualquier script

```
using UnityEngine;  
using System.Collections;
```

System.Collections pertenece a .NET

```
public class Example : MonoBehaviour  
{  
    // Use this for initialization  
    void Start ()  
    { ... }  
  
    // Update is called once per frame  
    void Update ()  
    { ... }  
}
```

Método *Start* se invoca por defecto para inicializar al objeto al arrancar el juego

Método *Update* se invoca por defecto en cada pasada del bucle principal del juego. Típicamente 60 fps

C#

Clases (II). Modificadores (I)

Static

Puede ser aplicado a atributos, métodos y clases

Una clase *static* sólo puede tener miembros *static* (variables y métodos)

Métodos y atributos *static* son miembros de clase y no miembros de objetos. Es decir

Son únicos y globales a la clase

Se pueden acceder sin crear objetos

```
static class weather
{
    enum TipoTiempo {Lluvioso, claro,
caluroso, nevado, frio, neblinoso, soleado};
}
```

```
class weather {
    enum TipoTiempo {Lluvioso, claro, caluroso,
nevado, frio, neblinoso, soleado};

    static TipoTiempo tiempo = TipoTiempo.caluroso;

    static TipoTiempo tiempoActual () { return tiempo; }

    static TipoTiempo predicciónTiempo() {
        TipoTiempo predicción = TipoTiempo.frio;
        //Realizar las acciones para predecir el tiempo
        //...

        return predicción;
    }
}
```


C#

Clases (III). Modificadores (II)

```
const          const float calorEspecificoAire = 1000;    //julios/Kg
```

Define valores constantes. Valores universales siempre: PI, RadsxVuelta

Son estáticos implícitamente

Su valor es conocido en tiempo de compilación

readonly

Sólo se puede asignar el valor inicial en la declaración o dentro del constructor. Equivalente a modificador *final* en Java

```
readonly float startTime = Time.time;
```

C#

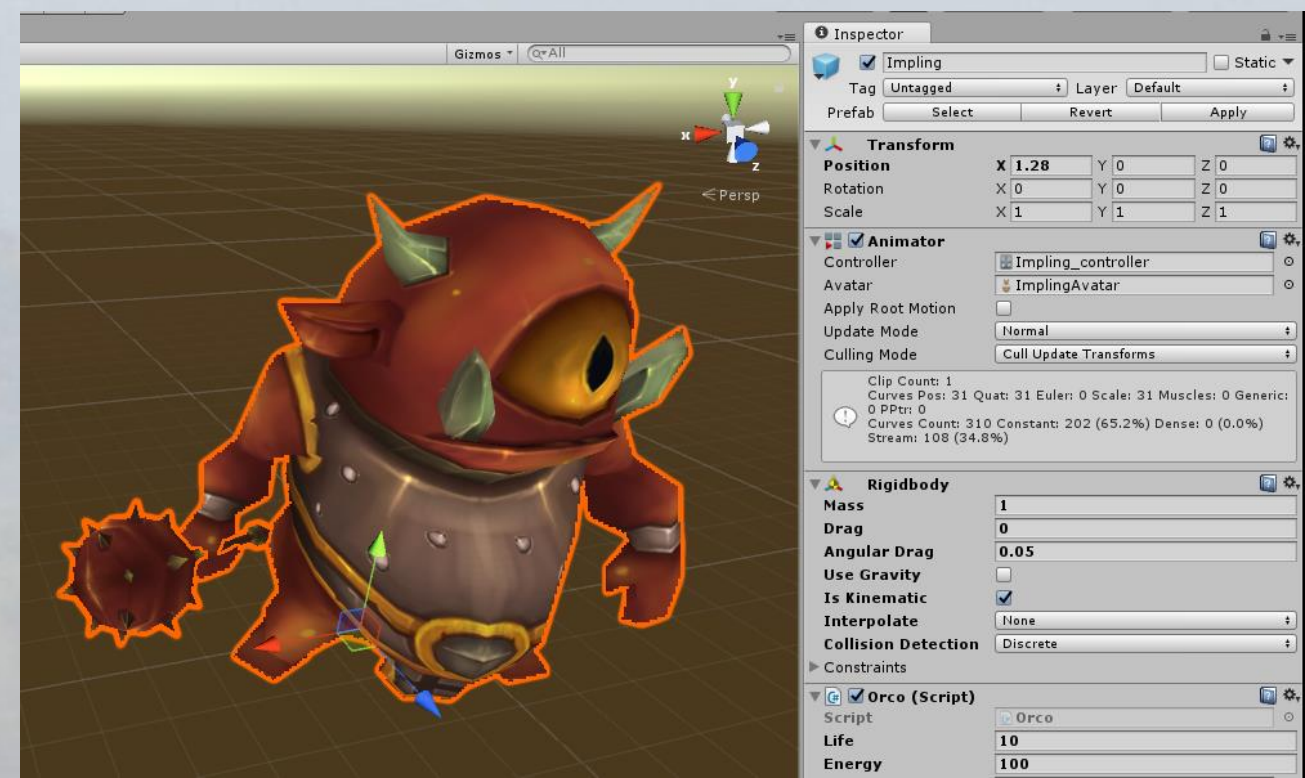
Ejemplo Clase (I)

Generar un nuevo proyecto Unity denominado Orco

Acceder a la tienda de contenidos de Unity y descargar al Orco Impling

<https://assetstore.unity.com/packages/3d/characters/creatures/3d-monster-0001-impling-53294>

Importarlo al proyecto como un asset



C#

Ejemplo Clase (II)

Generar una clase genérica denominada BasicChar en un juego que cualquier objeto del videojuego pueda almacenar su estado, vida,...

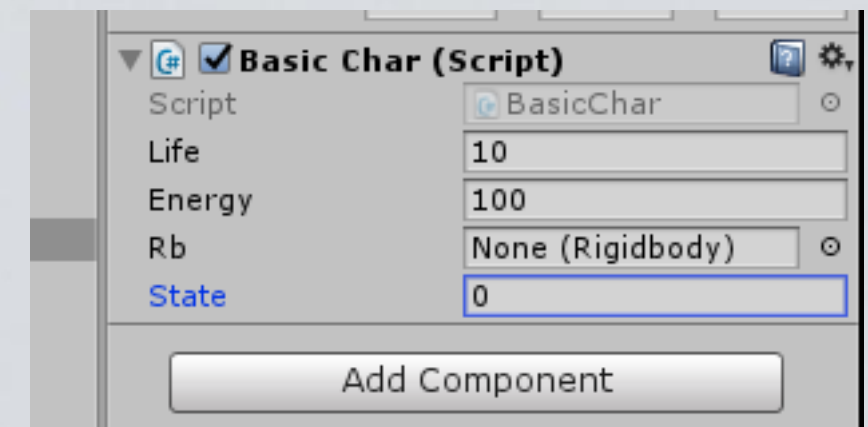
```
using UnityEngine;
using System.Collections;

public class BasicChar : MonoBehaviour {
    public const int idleState = 0,
                    movingState = 1,
                    diedState = 2;

    public float life = 10.0f,
                energy = 100.0f;

    public int state;

    public void init() {state = idleState;}
}
```



C#

Ejemplo Clase (III)


Ampliar la clase genérica para que pueda almacenar sus valores básicos físicos

```
using UnityEngine;
using System.Collections;

public class BasicChar : MonoBehaviour{

    public const int idleState = 0,
                    movingState = 1,
                    diedState = 2;


    public float life = 10.0f,
                energy = 100.0f;
    public Rigidbody rb;
    Vector3 posicionInicial,
            velocidadInicial,
            acc;
    public int state = idleState;
```



C#

Ejemplo Clase (III)

Ampliar la clase genérica para que pueda almacenar sus valores básicos físicos



```
public void init()
{
    rb = GetComponent<Rigidbody>();
    posicionInicial = transform.position;
    velocidadInicial = new Vector3(1.0f, 1.0f, 1.0f);
    acc = new Vector3(0.0f, 0.0f, 0.0f);
    state = idleState;
}


public void Reset()
{
    transform.position = posicionInicial;
    DetenerMovimiento();
    state = idleState;
}
```

C#

Ejemplo Clase (IV)

Ampliar la clase genérica para que pueda almacenar sus valores básicos físicos

```
public void DetenerMovimiento() {  
    rb.isKinematic = true;  
    rb.velocity = Vector3.zero;  
    acc = Vector3.zero;  
}  
  
//Kinetic update of the object  
public void kineticUpdate() {  
    Vector3 incVelocity = acc * Time.deltaTime,  
        avrgIncVelocity = incVelocity * 0.5f;  
  
    transform.position += (rb.velocity + avrgIncVelocity) * Time.deltaTime;  
    rb.velocity += incVelocity;  
}  
}
```



Bibliografía

Learning C# by Developing Games with Unity 3D Beginner's Guide. Terry Norton. Packt Publishing. ISBN 978-1-84969-658-6

Manuales en línea de Unity 3D



Documentación generada por
Dr. Ramón Mollá Vayá
Sección de Informática Gráfica
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia

Reconocimiento-NoComercial-CompartirIgual 2.5

Usted es libre de:

copiar, distribuir y comunicar públicamente la obra
hacer obras derivadas bajo las condiciones siguientes:



Reconocimiento. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador.



No comercial. No puede utilizar esta obra para fines comerciales.



Compartir bajo la misma licencia. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.

Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor

Los derechos derivados de usos legítimos u otras limitaciones reconocidas por ley no se ven afectados por lo anterior.