

Máster
V-ART

```
532 }
533 //StartCoroutine(IKBehaviour());
534 }
535 7 referencias
536 public void StateUpdate()
537 {
538     PlayerCollider.material = null;
539     if (!PlayerCollider.enabled)
540     {
541         PlayerCollider.enabled = true;
542     }
543     for (int i = 0; i < PlayerStatusObj.Length; i++)
544     {
545         PlayerStatusObj[i].SetActive(false);
546     }
547     if (StateParticle)
548     {
549         if (StateParticle.activeInHierarchy)
550         {
551             StateParticle.transform.position = this.gameObject.transform.position;
552             StateParticle.GetComponent<ParticleSystem>().Play();
553         }
554     }
555 }
```

Master en: Programación y desarrollo de videojuegos

480
HORAS

- El videojuego es un arte que lleva muchos años entre nosotros, aportando nuevos tipos de entretenimiento a la par que enseñanzas y experiencias nunca vistas antes.
- En este Máster descubrirás todas las herramientas necesarias para introducirte y profesionalizarse en el desarrollo de videojuegos con Unity. Aprenderás diferentes técnicas para poder desarrollar tus proyectos de una manera eficiente y poder transmitir emociones a otras personas a través de este medio.
- Al terminar el curso serás capaz de enfrentarte a problemas reales de un desarrollo completo en Unity y poder participar en un proyecto de videojuego con total soltura.



1. INTRODUCCIÓN A UNITY

- Usos comunes.
- ¿Por qué Unity?.
- Aplicaciones de Unity para diferentes mercados/sectores.
- Instalación y uso de Unity Hub.
 - Instalación de Unity.
- Configuración inicial.

2. VENTANAS E INTERFAZ EN UNITY

- Ventanas por defecto de Unity.
- Shortcuts comunes de Unity.
 - Project Settings.
- Ventana de Preferences

3. TRATAMIENTO DE ASSETS EN UNITY

- Ventanas de importación de Assets 2D
- Ventanas de importación de assets 3D
- Exportación de paquetes personalizados.

4. JERARQUÍA DE PROYECTO EN UNITY

- Estructura de proyecto.
- Organización de assets en el proyecto.
- Organización de objetos en la escena.

5. INTRODUCCIÓN A PROGRAMACIÓN EN C#

- Variables
- Datos primitivos
- Uso y tratamiento de variables
 - Métodos y funciones

6. PROGRAMACIÓN ORIENTADA A OBJETOS EN UNITY

- *Monobehaviour*
- *GameObjects*
- *Componentes*

7. MOVIMIENTO DE PERSONAJE CON CHARACTER CONTROLLER

- Configuración del Character Controller
- Programación y uso del Character Controller

8. USO DE CÁMARAS EN UNITY

- Overview del componente *Camera*.
- Creación de una cámara con movimiento.
- Overview del paquete *Cinemachine*.

9. CREANDO EL GAME MANAGER

- Creación de variables de control global.
- Creación de sistema de control de juego.

10. SCRIPTS EN UNITY

- Orden de ejecución de scripts.
- Métodos comunes de *Monobehaviour*.

11. TRABAJANDO CON INTERFACES EN UNITY

- ¿Qué es el Canvas?
- Creación y programación de un botón.
- Actualización a tiempo real de la interfaz de usuario.

```
(Global Scope)
124 // Retrieve the current depth value of the surface behind the
125 // pixel we are currently rendering.
126 float existingDepth01 = tex2Dproj(_CameraDepthTexture, UNITY_PROJ_COORD(i.screenPosition)).r;
127 // Convert the depth from non-linear 0..1 range to linear
128 // depth, in Unity units.
129 float existingDepthLinear = L.LinearEyeDepth(existingDepth01);
130
131 // Difference, in Unity units, between the water's surface and the object behind it.
132 float depthDifference = existingDepthLinear - i.screenPosition.w;
133
134 // Calculate the color of the water based on the depth using our two gradient colors.
135 float waterDepthDifference01 = saturate(depthDifference / _DepthMaxDistance);
136 float4 waterColor = lerp(_DepthGradientShallow, _DepthGradientDeep, waterDepthDifference01);
137
138 // Retrieve the view-space normal of the surface behind the
139 // pixel we are currently rendering.
140 float3 existingNormal = tex2Dproj(_CameraNormalsTexture, UNITY_PROJ_COORD(i.screenPosition));
141
142 // Modulate the amount of foam we display based on the difference
143 // between the normals of our water surface and the object behind it.
144 // Larger differences allow for extra foam to attempt to keep the overall
145 // amount consistent.
146 float3 normalDot = saturate(dot(existingNormal, i.viewNormal));
147 float foamDistance = lerp(_FoamMaxDistance, _FoamMinDistance, normalDot);
148 float foamDepthDifference01 = saturate(depthDifference / foamDistance);
149
150 float surfaceNoiseCutoff = foamDepthDifference01 * _SurfaceNoiseCutoff;
151
152 float2 distortSample = (tex2D(_SurfaceDistortion, i.distortUV).xy * 2 - 1) * _SurfaceDistortionAmount;
153
154 // Distort the noise UV based off the RG channels (using xy here) of the distortion texture.
155 // Also offset it by time, scaled by the scroll speed.
156 float2 noiseUV = float2((i.noiseUV.x + _Time.y * _SurfaceNoiseScroll.x) + distortSample.x,
157 (i.noiseUV.y + _Time.y * _SurfaceNoiseScroll.y) + distortSample.y);
158 float surfaceNoiseSample = tex2D(_SurfaceNoise, noiseUV).r;
159
160 // Use smoothstep to ensure we get some anti-aliasing in the transition from foam to surface.
161 // Uncomment the line below to see how it looks without AA.
162 float surfaceNoise = surfaceNoiseSample > surfaceNoiseCutoff ? 1 : 0;
```

12. CREACIÓN DE TERRENOS

- Uso de la herramienta Terrain de Unity.
- Bases de creación de terrenos en Unity.

13. OPTIMIZACIÓN EN UNITY

- Generación de lightmaps (Bake)
 - Cullings

14. MANEJANDO ESCENAS EN UNITY

- Creación de escenas.
- Programación de carga de escenas en C#.
 - Cargas aditivas y pantallas de carga.

15. TRABAJANDO ANIMACIONES EN UNITY

- Animator
- Animation
- Creación de Keyframe animations

16. SISTEMAS EN AUDIO EN UNITY

- Audio Source.
- Reproducir sfx específicos.
- Programar un sistema de Audio en C#.

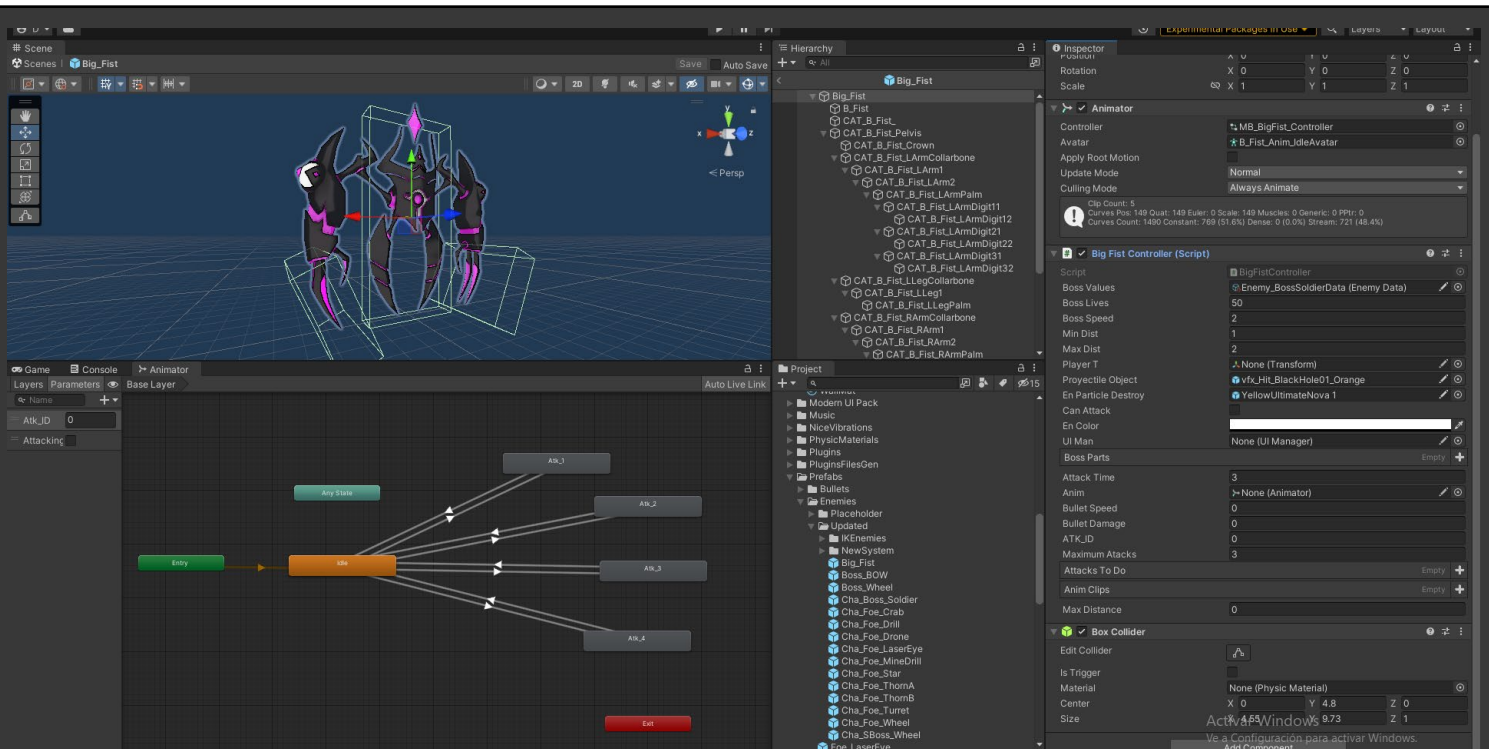


17. SERIALIZADO Y GUARDADO DE DATOS EN UNITY

- Diferentes métodos de guardado de datos en Unity.
- Player Prefs.

18. PROTOTIPO DE PROYECTO EN UNITY

- Introducción al diseño de niveles.
- Introducción al Game Design.
- Herramientas comunes para blocking en Unity.
- Estructuras y diseño de blocking en Unity.



19. EXPORTACIÓN DE PROYECTOS

- Configuración para exportación.
- Diferentes tipos de compilación
 - Assemblies

PROYECTO JUEGO DE PLATAFORMAS 2D (24H)

20. PROGRAMACIÓN EN UNITY

- Coroutines.
- Sistemas de eventos.
 - Singleton Pattern.
 - Observer Pattern.

21. MULTIJUGADOR LOCAL EN UNITY

- Input system
- Programación y uso de diferentes inputs.

21. SCRIPTABLE OBJECTS

- Creación y uso.
- Diferentes aplicaciones.

23. PERSONALIZANDO EL EDITOR DE UNITY

- Librerías de Editor
- Creación de botones propios en la interfaz de Unity.
- Creación de ventanas personalizadas en Unity.

24. RENDER PIPELINES

- URP
- HDRP
- SRP

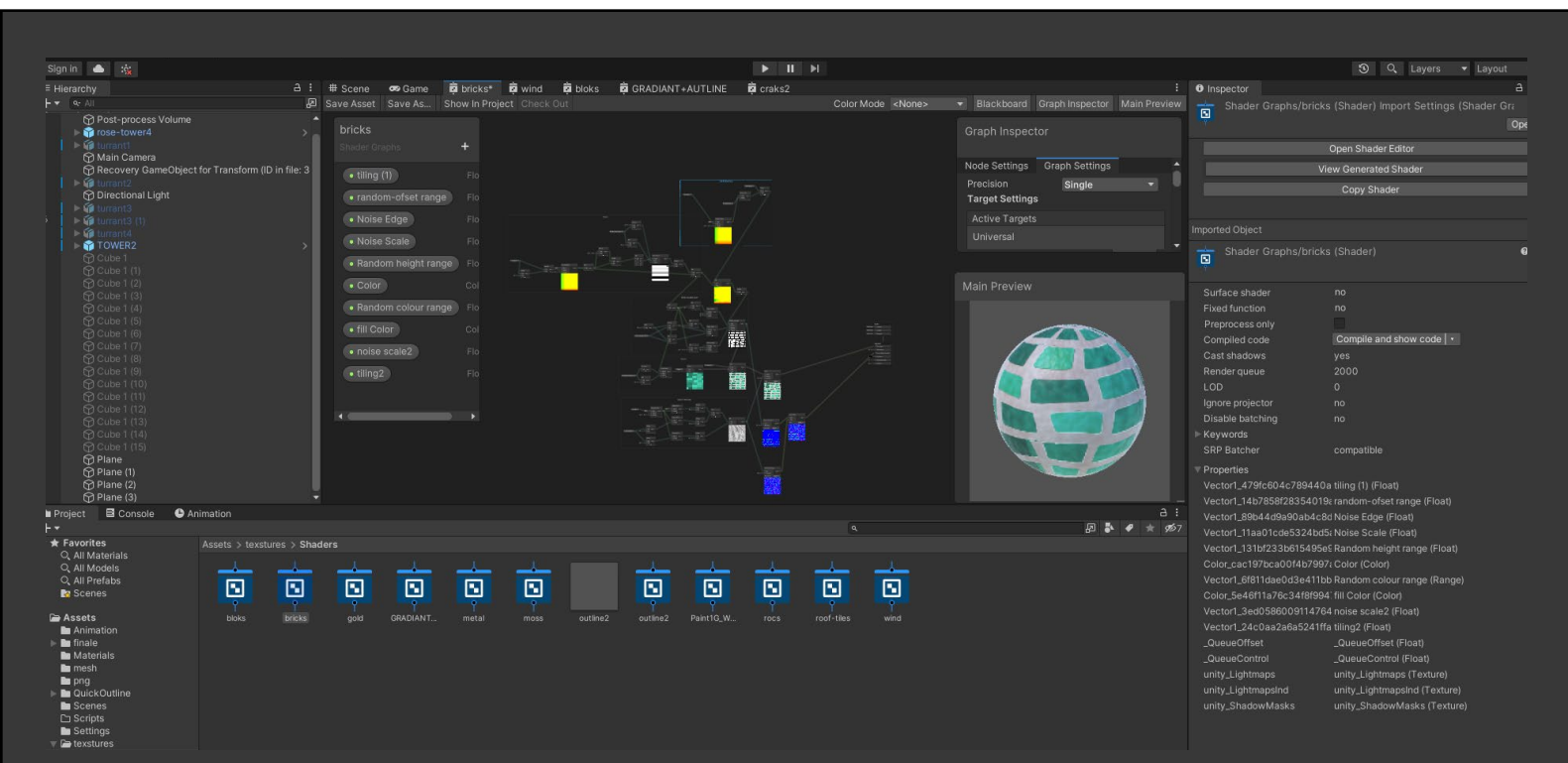
25. PROGRAMANDO SHADERS

- Introducción a Shader Graph
- Creación de un Shader lit básico
- Creación de un Shader de agua.

PROYECTO SHOOTER 3D (24H)

26. ASSETS BUNDLES

- Instalación y bases.
- Lectura y ejecución de Assets Bundles.



27. CREANDO HERRAMIENTAS PERSONALIZADAS

- Creación de sistema de eventos personalizados en Editor.
- Creación de herramienta para compilación de Editor personalizada.

28. OPTIMIZACIÓN EN UNITY II

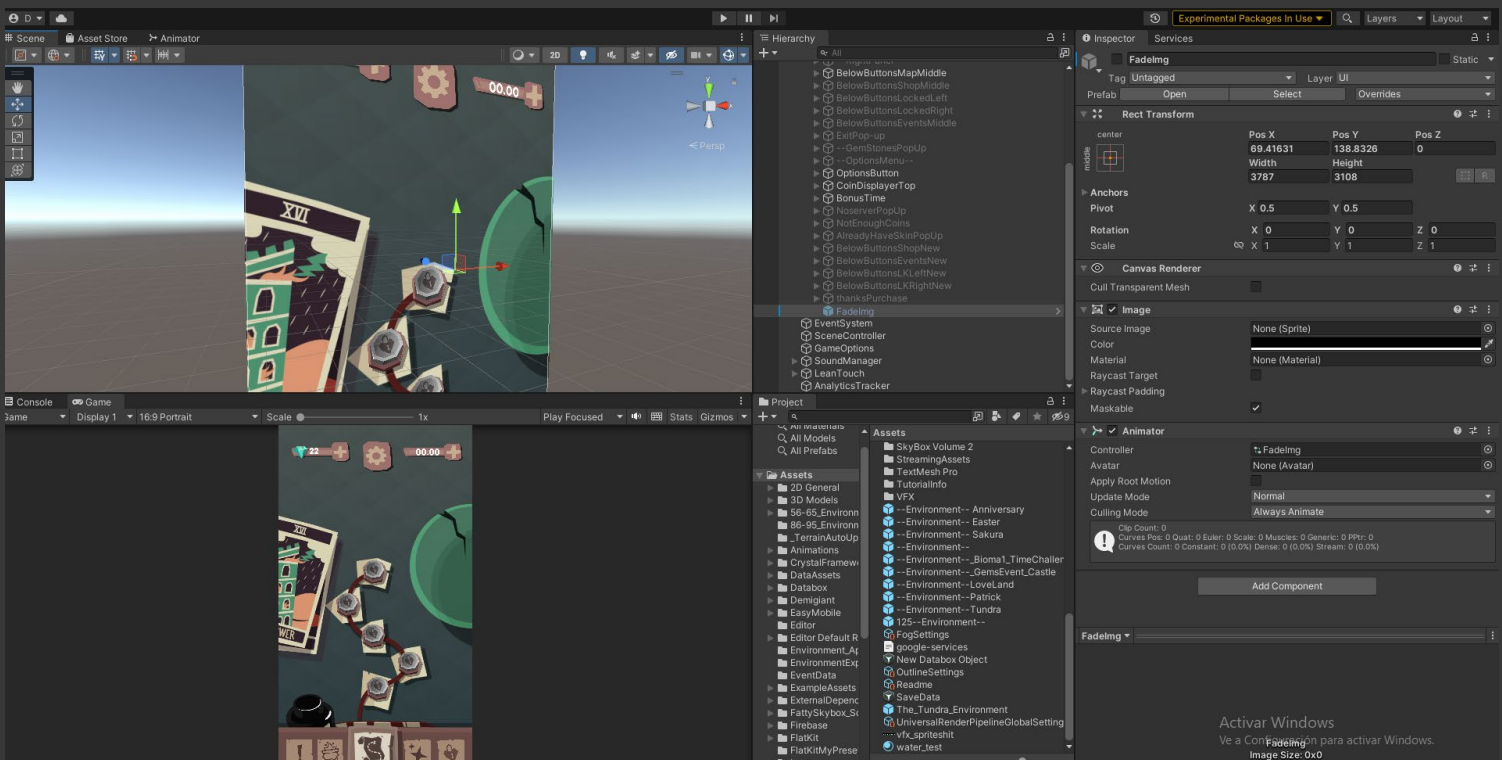
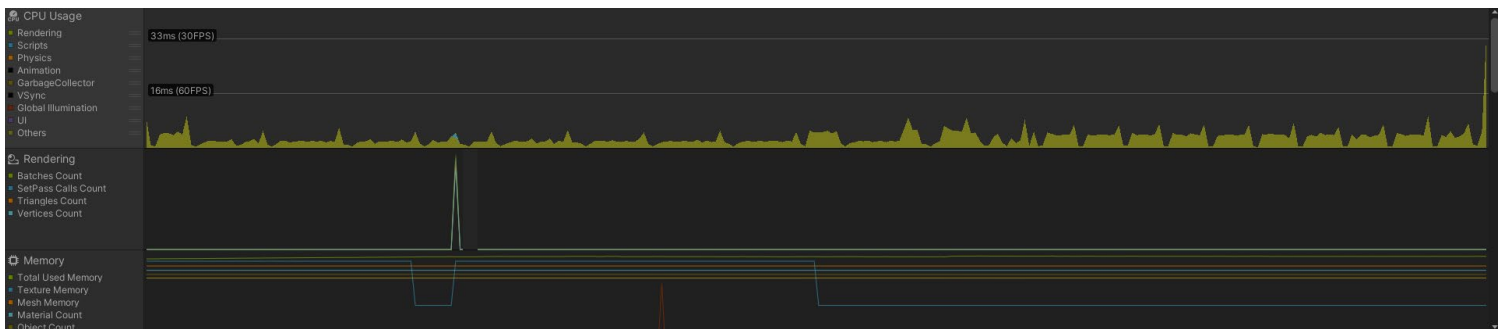
- Shader Stripping.
- Render Scales.
- Profiler.

29. UNITY MOBILE

- Librerías de android
- Configuración de Unity para Android
- Diseño de videojuego para móvil.
- Input System orientado a móvil
- Optimización para móvil.

PROYECTO

PLATAFORMAS PARA MÓVIL (24H)



30. REALIDAD VIRTUAL CON UNITY

- Unity orientado a Realidad Virtual.
- Librerías de Unity para Realidad Virtual.
 - Configuración para Realidad Virtual
- Game Design orientado a Realidad Virtual.

PROYECTO SHOOTER VR (24H)

31. REALIDAD AUMENTADA CON UNITY

- Unity orientado a Realidad Aumentada.
- Librerías de Unity para AR
- Configuración para Realidad Aumentada.
- Game Design orientado a Realidad Aumentada.

PROYECTO CLICKER AR (24H)

PROYECTO FINAL

