AÑADIR MANOS PERSONALIZADAS

Reemplaza los controladores que vienen por defecto por unas manos de verdad. Puedes crear tus propias manos fácilmente, a partir de un modelo 3D y el sistema de animación de Unity que posiblemente ya conoces, pero también puedes importar en tu proyecto de Unity un paquete ya preparado que hay disponible en Google Drive:

https://drive.google.com/drive/folders/1F4C1VppOTOm1eJ4lVMCjD-Ll-LCn4A3s?usp=drive_link

Mira el contenido del paquete y verás que consiste en dos prefabs con los modelos 3D y las animaciones de cada una de las manos.

Selecciona en la jerarquía Left Controller y ve a las opciones del Inspector > XR Controller > la última opción es Model > Model Prefab. Arrastra ahí el prefab de Left Hand Model. Haz lo mismo con la mano derecha.

Vamos a ver qué animaciones hay en el prefab de cada mano. Para ello, seleccionamos el prefab y vemos que tiene asociado un componente Animator. Para ver la herramienta de edición de Animator, vamos a Window->Animation->Animator

Pinchando en Blend Tree (primero dos veces para ver su interior, y luego una vez en la nueva vista) podemos ver las animaciones sobre el propio modelo de la mano:

Vemos que hay dos parámetros de animación, Grip y Trigger que, según los modifiquemos cambiarán la pose de la mano de manera continua. El parámetro Trigger va ligado a la pulsación del Trigger del mando Touch con el dedo índice, y el parámetro Grip va ligado a la pulsación del Grip con el dedo anular (lo cual habitualmente se interpreta como cerrar el puño).

Estos valores son un float entre 0 y 1 que podremos controlar fácilmente desde script, así que solo necesitamos leer los controles del mando y asignar su valor a estos parámetros.

Añade al prefab de las manos el siguiente script:

```
using UnityEngine.InputSystem;

public class MyHandController : MonoBehaviour
{
    [SerializeField] InputActionReference actionGrip;
    [SerializeField] InputActionReference actionTrigger;

    private Animator handAnimator;

    void Start()
    {
        actionTrigger.action.performed += TriggerPress;
        actionGrip.action.performed += GripPress;
        handAnimator = GetComponent<Animator>();
```

```
    }

    private void GripPress(InputAction.CallbackContext obj)
    {
        handAnimator.SetFloat("Grip", obj.ReadValue<float>());
    }

    private void TriggerPress(InputAction.CallbackContext obj)
    {
        handAnimator.SetFloat("Trigger", obj.ReadValue<float>());
    }
}
```

En este caso se utiliza otra forma de referenciar una InputAction concreta. En lugar de pedir al usuario que especifique un InputActionAsset para buscar dentro un nombre concreto, permitimos al usuario que especifique cualquier InputAction desde la ventana del Inspector gracias a la clase InputActionReference.

Si abres el editor del InputActionAsset, verás que en cada mano hay una acción predefinida llamada Select Value.

Esta acción está ligada al control Grip del mando. A diferencia del Primary Button del ejemplo anterior, esta acción es de tipo Value/Axis. Esto es debido a que el control Grip es de tipo continuo y nos da un valor entre 0 (sin pulsar) y 1 (totalmente pulsado).

Por tanto, podemos usar esta acción para el control de la animación de Grip en nuestro script. Para la animación de Trigger tenemos la acción Activate Value (compruébalo en el editor del asset).

¡Ejecuta ahora la escena para ver tus manos animadas!

Intenta hacer esto mismo pero utilizando en tu script el método Update en lugar de crear nuevas callbacks.