VR Shooter Kit 1.3

Hola y gracias por tu compra de **VR Shooter Kit.**

Si necesitas algún tipo de asistencia, ayuda o tienes alguna idea para mejorar este asset, escribeme a **james@unity3dninja.com**, y estaré feliz de ayudar.

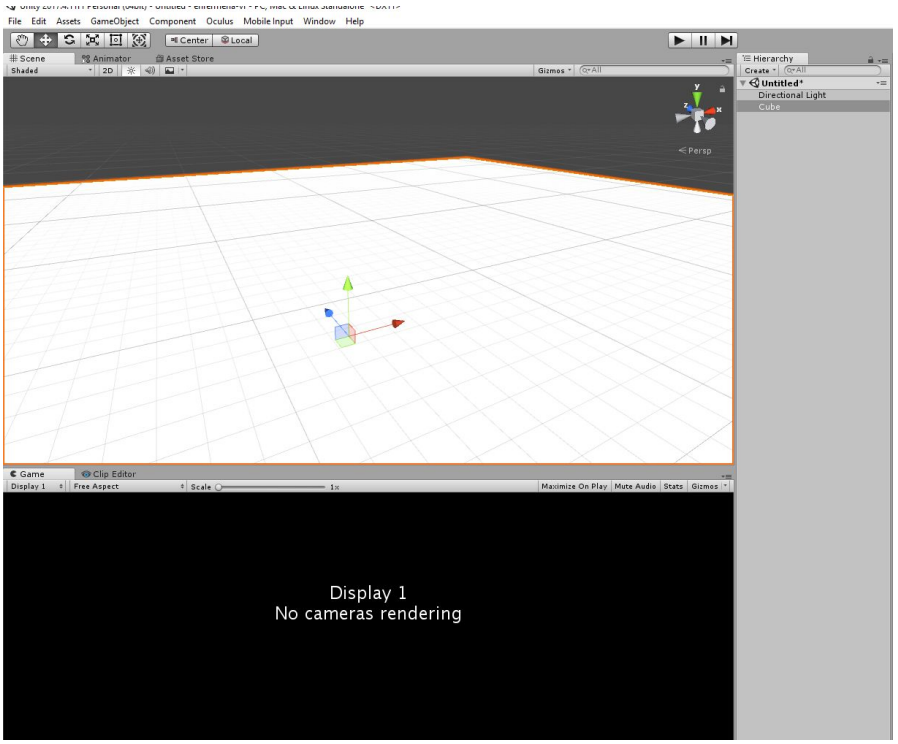
VR Shooter Kit es un asset que busca facilitar el desarrollo de juegos VR, con una serie de scripts y ejemplos, para que puedas construir tus propios juegos.

Todo el código incluido esta comentado así que si quieres entender cómo funciona solo debes abrir cualquier script y empezar a leer un poco, aunque en este documento trataré de explicar las partes más relevantes del mismo.

Inicio Rápido

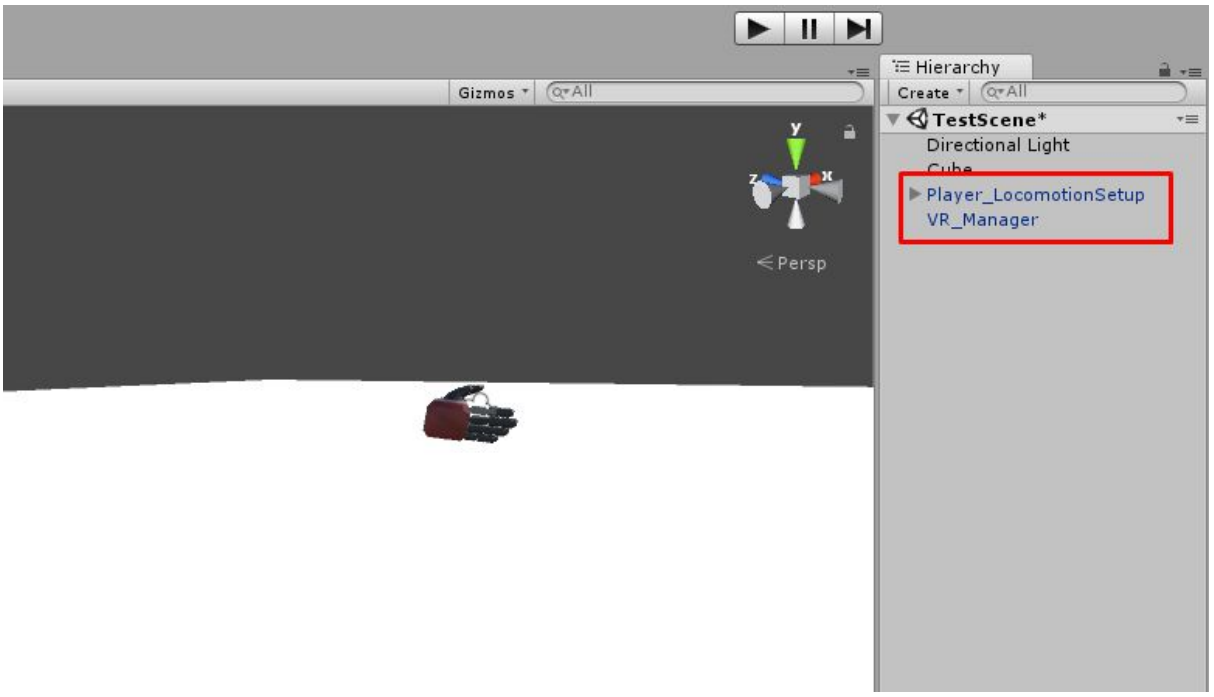
Lo más recomendable es que empieces a jugar con la escena demo, para que puedas familiarizarte un poco en este asset.

Si quieres crear tu propio juego, solo debes crear una nueva escena y asegurarte de que no haya ninguna cámara.



Yo he creado un cubo que servirá como el suelo, luego debes arrastrar un prefab para el jugador , puedes escoger **Player_LocomotionSetup** o **Player_TeleportSetup**, desde la carpeta **VRShooterKit/Prefabs/PlayerSetup.**

Luego solo debes arrastrar el **VR_Manager** desde **VRShooterKit/Prefabs/Main.**



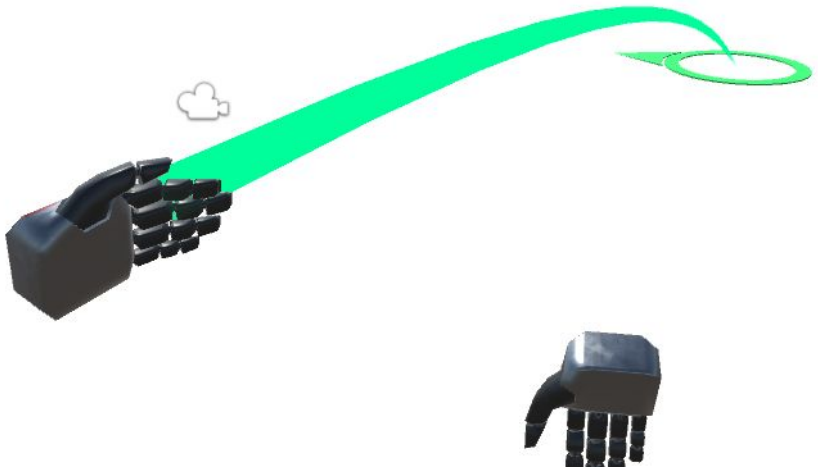
y tan solo con esto ya estamos listos.

Controles Player_LocomotionSetup

El player locomotion setup, usa el **OVRPlayerController.cs** del **Oculus Integration,** así que los controles aquí son los estándares, con el control izquierdo puedes moverte alrededor y con el derecho puedes girar, con el botón trigger del control izquierda puedes correr.

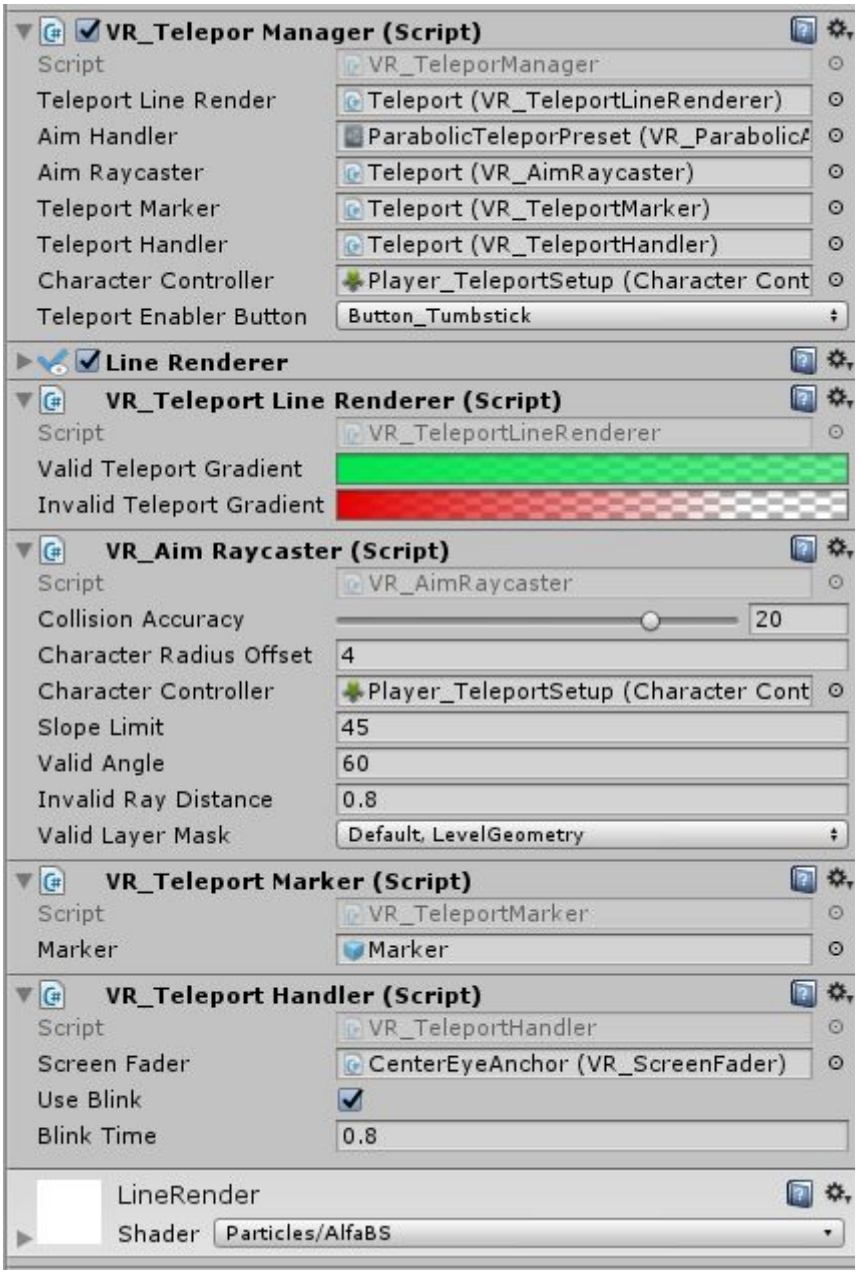
Controles Player_TeleportSetup

El player teleport setup, usa la mecánica de teleport que podrías ver en juegos como robo recall o similares, puedes teletransportarte con cualquier control solo basta con mover el joystick, y podrás ver la línea junto con el marcador, que te indican tu posición final, para completar el movimiento basta con llevar el joystick a su posición inicial.



Puedes mover el joystick alrededor para indicar la rotación que quieres, y mover tu mano para indicar la posición.

El sistema de teletransporte es bastante configurable y puede ser fácilmente extendido, basta con seleccionar el objeto teleport para ver todas las opciones que tenemos.



VR_Grabbable.cs

Quizás el script más importante en este asset, como seguramente ya sabes puedes añadir este script a cualquier objeto que el jugador pueda tomar.

bool SetJointSettings: indica si queremos configurar el joint usado para agarrar objetos, actualmente se usa un FixedJoint, habilita los campos **JointBreakTorque** y **JointBreakForce.**

(optional) float JointBreakTorque: Indica la fuerza que debe aplicar para romper el joint.
(optional) float JointBreakForce: Indica la fuerza que debe aplicar para romper el joint.

float InteractDistance: Indica la distancia a la que se puede tomar este objeto.

bool PerfectGrab: indica si el objeto usa perfect grab, es decir, el objeto no se moverá al punto de agarre, habilita los campos **ShouldFly** y **AutoGrab.**

bool UsePerHadSettings: Indica si queremos usar configuraciones diferentes para cada mano, habilita los campos **RightHandSettings** y **LeftHandSettings.**

VR_HandInteractSettings HandSettings: configuración global para ambas manos.

- **Transform InteractPoint:** Punto desde el cual se agarra el objeto.
- **Transform HighlightPoint:** Punto en el cual indicara que la mano puede tomar el objeto ,usando el **VR_OutlineHighlight** o **VR_UIHighlight.**
- **Vector3 RotationOffset:** Indica el offset de la rotation, al tomar el objeto.
- **AnimationClip Animation:** animacion para la mano cuando tome elobejto.
- **bool CanInteract:** indica si esta mano puede tomar el objeto.
- **bool HideHandOnGrab:** Indica si la mano debe ocultarse al tomar el objeto.

(optional)VR_HandInteractSettings RightHandSettings: configuracion para la mano derecha.

(optional)VR_HandInteractSettings LeftHandSettings: configuración para la mano izquierda.

bool ShouldFly: Indica si al tomarse el objeto este debe de moverse hacia la mano, habilita el campo **GrabFlyTime.**
(optional) float GrabFlyTime: indica el tiempo que tardará el objeto en llegar a la mano.

(optional) bool AutoGrab: indica si este objeto debe agarrarse automaticamente, habilita los campos **StartOnRightHand** y **StartOnLeftHand.**

(optional) button StartOnRightHand: indica si el objeto debe empezar en la mano derecha.

(optional) button StartOnLeftHand: indica si el objeto debe empezar en la mano izquierda.

bool EnableColliderOnGrab: Indica si al momento de agarrar el objeto, queremos tener el collider activado.
VR_Button InteractButton: botón que se debe presionar para iniciar la interacción.

int GrabLayer: Indica el layer a utilizar cuando se agarra un objeto.
int UnGrabLayer: Indica el layer a utilizar cuando se suelta el objeto.

bool preserveKinematicState: Indica si el estado kinematico del objeto debe cambiar, cuando se suelta.

UnityEvent OnGrabStateChangeEvent: Evento que se llama cuando el estado de agarre del objeto cambia.

Para crear un objeto que se pueda agarrar, basta solo con añadir este script a cualquier objeto.

VR_Weapon.cs

Este script te permite crear armas dentro del juego.

enum ReloadMode ReloadMode: Indica la forma en la que se procesa la recarga de las armas.

- **ReloadMode.Physics:** Indica que el arma debe recargarse mediante algún gesto, como el revólver en escena de ejemplo.
- **ReloadMode.Realistic:** Indica que el arma debe recargarse cambiando el cargador del arma, por uno que aún posea munición.
- **ReloadMode.UI:** Indica que el arma se recargara automáticamente, mostrando una pequeña barra indicado el tiempo que tardará en recargar.
- **ReloadMode.InfiniteBullets:** Indica que el arma jamas necesita ser recargada.

(optional) WeaponUI WeaponUI: Componente usado para indicar la cantidad de munición restante, y la barra de carga que solo se usa en **ReloadMode.UI.**

(optional) BarrelScript BarrelScript: Componente usado en la animación de recarga del revolver, solo requerido en **ReloadMode.Physics**.

(optional) float ReloadAngle: Ángulo en el que debe terminar la mano para iniciar una recarga, solo usado en **ReloadMode.Physics.**

(optional) enum WeaponTag WeaponTag: Enum que indica que tipo de recarga puede aceptar esta arma, solo requerido en **ReloadMode.Realistic**.

(optional) Transform MagazineSnapPoint: Punto en el cual se coloca la recarga, solo requerido en **ReloadMode.Realistic**.

(optional) int ClipSize: Tamaño del cargador, solo requerido en **ReloadMode.UI**.

(optional) float ReloadTime: tiempo que tarda en realizarse la recarga, solo requerido en **ReloadMode.UI**.

Transform ShootPoint: punto desde el cual se dispararan las balas.
Bullet BulletPrefab: bala usada por el arma.
WeaponHammer WeaponHammer: componente que simula el uso de un martillo en arma.
ShellEjector ShellEjector: Componente que maneja la expulsión de casquillos en el arma.
float ShootRate: indica el tiempo que debe pasar entre cada disparo.
bool IsAutomatic: indica si el arma es automática.
float bulletSpeed: velocidad de la bala al ser disparada.
int HitLayer: indica cuales layers puede golpear la bala.
int MaxBulletBounceCount: indica cuántas veces puede rebotar una bala en una superficie.

float Dmg: indica el daño que produce cada bala.

float MinHitForce: impacto mínimo aplicado al objeto que golpea la bala.
float MaxHitForce: impacto máximo aplicado al objeto que golpea la bala.
float Range: distancia máxima que puede viajar la bala.
AudioClip ShootSound: Sonido reproducido al momento de disparar.
GameObject MuzzleFlash: Efecto del disparo.
bool ParentMuzzleFlash: Indica si el efecto de disparo debe ser hijo del shootPoint:
bool DisableMuzzleWhileNoShooting: Indica si el efecto de disparo debe ser desactivado mientras no se dispara.
VR_Button FireButton: botón que inicia la acción de disparo.

float MinRecoilPositionForce: Indica la fuerza mínima de movimiento, que ejerce el arma al dispararse.
float MaxRecoilPositionForce: Indica la fuerza máxima de movimiento, que ejerce el arma al dispararse.
float RecoilPositionLimit: fuerza máxima acumulada de movimiento, que puede ejercer el arma.

float MinRecoilRotationForce: Indica la fuerza mínima de rotación, que ejerce el arma al dispararse.
float MaxRecoilRotationForce: Indica la fuerza máxima de rotación, que ejerce el arma al dispararse.
float RecoilAngleLimit: Ángulo máximo acumulado, que puede ejercer un arma al dispararse.

bool UseSpread: Indica si la bala debe fragmentarse, normalmente utilizado en las escopetas, habilida los campos **MinSpreadCount, MaxSpreadCount, MinSpreadAngle y MaxSpreadAngle**

int MinSpreadCount: cantidad mínima de fragmentos.
int MaxSpreadCount:cantidad máxima de fragmentos.
float MinSpreadAngle: ángulo mínimo de trayectoria que tendrá cada fragmento.
float MaxSpreadAngle: ángulo mínimo de trayectoria que tendrá cada fragmento.

float PositionLerpSpeed: valor aplicado para retornar a la posición inicial.
float RotationLerpSpeed: valor aplicado para retornar a la rotación inicial.

VR_Lever

float InteractDistance: Indica la distancia a la que se puede tomar este objeto.
bool UsePerHadSettings: Indica si queremos usar configuraciones diferentes para cada mano, habilita los campos **RightHandSettings** y **LeftHandSettings.**

VR_HandInteractSettings HandSettings: configuración global para ambas manos.

- **Transform InteractPoint:** Punto desde el cual se agarra el objeto.
- **Transform HighlightPoint:** Punto en el cual indicara que la mano puede tomar el objeto ,usando el **VR_OutlineHighlight** o **VR_UIHighlight.**
- **Vector3 RotationOffset:** Indica el offset de la rotation, al tomar el objeto.
- **AnimationClip Animation:** animacion para la mano cuando tome elobejto.
- **bool CanInteract:** indica si esta mano puede tomar el objeto.
- **bool HideHandOnGrab:** Indica si la mano debe ocultarse al tomar el objeto.

(optional)VR_HandInteractSettings RightHandSettings: configuracion para la mano derecha.

(optional)VR_HandInteractSettings LeftHandSettings: configuración para la mano izquierda.

VR_Button InteractButton: botón que se debe presionar para iniciar la interacción.

Transform TransformBase: base de la palanca para ignorar colisiones, y obtener la dirección.

int SolverIteractions: Indica el número de **SolverIteractions**, más información aquí.

bool ShouldBackToStartingPosition: Indica si la palanca de volver a su posición inicial.
float BackForce: fuerza con la cual la palanca regresa a su posición inicial.

UnityEvent OnValueChange: Evento que se llama cuando la palanca cambia de posición, 0 indica que la palanca está en posición inicial, 1 que la palanca está en su posición final.

UnityEvent OnGrabStateChangeEvent: Evento que se llama cuando el estado de agarre del objeto cambia.

VR_DropZone

enum DropZoneMode DropZoneMode: Indica el modo en el que funciona el **VR_DropZone.**

- **VR_DropZoneMode.Collider:** los objetos que entren en contacto con los collider, activaran el dropzone.
- **VR_DropZoneMode.Distance:** los objetos que se encuentren en el rango, activaran el dropzone.

Transform DropPoint: Indica la posicion y rotacion final, de los objetos que entren en el DropZone.

VR_Grabbable StartingDrop: indica si hay algún objeto inicialmente en el DropZone.

Collider[] DropZoneColliderArray: Indica los collider que usará el DropZone.
float DropRadius: Indica la distancia mínima en la que los objetos pueden activar el DropZone.

bool ShouldFly: Indica si el objeto debe moverse hacia el **DropPoint.**
bool SyncronizePosition: Indica si el DropZone debe modificar la posición de los objetos.
bool SyncronizeRotation: Indica si el DropZone debe modificar la rotación de los objetos.
bool UsePreview: Indica si el DropZone usará un preview.

UnityEvent OnDropStateChange: Evento llamado cuando un objeto es removido,o añadido en el DropZone.

VR_Button

List<Collider> IgnoreColliderList: Indica que colliders debe ignorar el botón.
Transform ObstacleCastPoint: Punto desde el cual este botón debe buscar por obstáculos.
float ObstacleCastRadius: Radio en el cual se debe buscar por obstáculos.
float PressThreshold: distancia que este botón debe recorrer para alcanzar su posición final.
float PressTime: Tiempo que tarda este botón, en llegar a la posición de presionado.
Vector3 PressingDir: Dirección hacia la cual se presiona el botón.