**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| PROGRAMA DE FORMACIÓN | Desarrollo de videojuegos y entornos interactivos |
| --- | --- |

| COMPETENCIA | 220501088- Producir la lógica del videojuego de acuerdo con el diseño y técnicas de desarrollo. | RESULTADOS DE APRENDIZAJE | 220501088-2. Construir el prototipo funcional del videojuego siguiendo la estructura de código. |
| --- | --- | --- | --- |

| NÚMERO DEL COMPONENTE FORMATIVO | CF009 |
| --- | --- |
| NOMBRE DEL COMPONENTE FORMATIVO | Programación de eventos de personajes y entornos |
| BREVE DESCRIPCIÓN | El componente formativo se enfoca en el diseño de interfaces de usuario y eventos para facilitar la interacción de las diferentes acciones que ocurrirán en los niveles de juego. |
| PALABRAS CLAVE | Canvas clase, event, evento, UI |

| ÁREA OCUPACIONAL | 5 - Arte, cultura, esparcimiento y deportes |
| --- | --- |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

**Introducción**

* 1. **Programación de comportamientos de personajes controlados por el usuario**
  2. Configuración de animaciones de los personajes
  3. Máquina de estado finito
  4. **Programación de eventos de interfaz HUD**
  5. Componentes del Canvas
  6. Interfaces de usuario responsivas
  7. Eventos

1. **Programación de eventos disparadores (interacción con el entorno)**

4. **Programación de comportamientos relativos a la mecánica de videojuego**

**5. Programación de inteligencia artificial**

1. **INTRODUCCIÓN**

Los contenidos desarrollados en este componente formativo están enfocados en el diseño de interfaces de usuario y la programación de eventos y personajes para facilitar la interacción de las diferentes acciones que se dan en un videojuego.

Se invita a ver el video a continuación para conocer un poco más sobre los contenidos a desarrollar en este componente:

| Video  Ver anexo: CF09\_Video\_Introduccion |
| --- |

1. **DESARROLLO DE CONTENIDOS**
   * 1. **Programación de comportamientos de personajes controlados por el usuario**

La programación de los comportamientos de los personajes permitirá definir las diferentes acciones que serán ejecutadas por el personaje cuando el jugador interactúe con los diferentes dispositivos de juego (*joystick*, teclado, mouse).

**Input manager**

El input manager es una ventana especializada en el motor de desarrollo Unity3D que permite definir los diferentes ejes (*axes*) que se encuentren asociados a las diferentes acciones del juego y/o personaje.

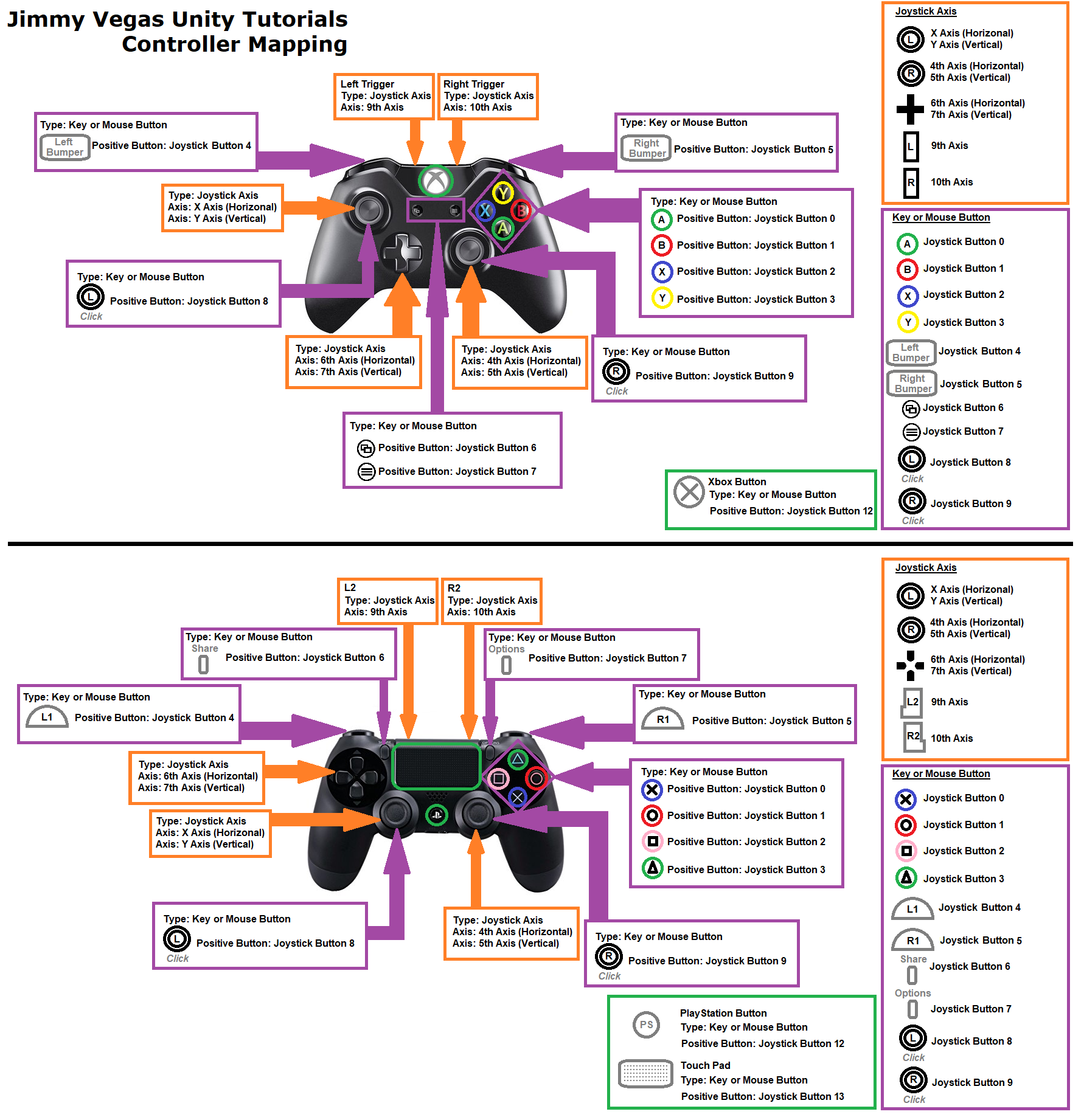
El input manager en Unity3D utiliza los siguientes tipos de controles:

| Controles Unity3D | |
| --- | --- |
| Key | Hace referencia a cualquier tecla (*key*) que se encuentre en teclado físico, como por ejemplo X, Ctrl, Shift. |
| Button | Hace referencia a cualquier tipo de botón que se encuentre en un control físico como un *gamepad, joystick.* |
| Virtual Axis | Permite realizar el mapeo de los ejes de un control físico. Por ejemplo, cuando el jugador mueve las palancas de un *gamepad*. |

En la siguiente figura se puede observar el mapeo de controles de Unity:

**Figura 1**

*Mapeados controles de juego*

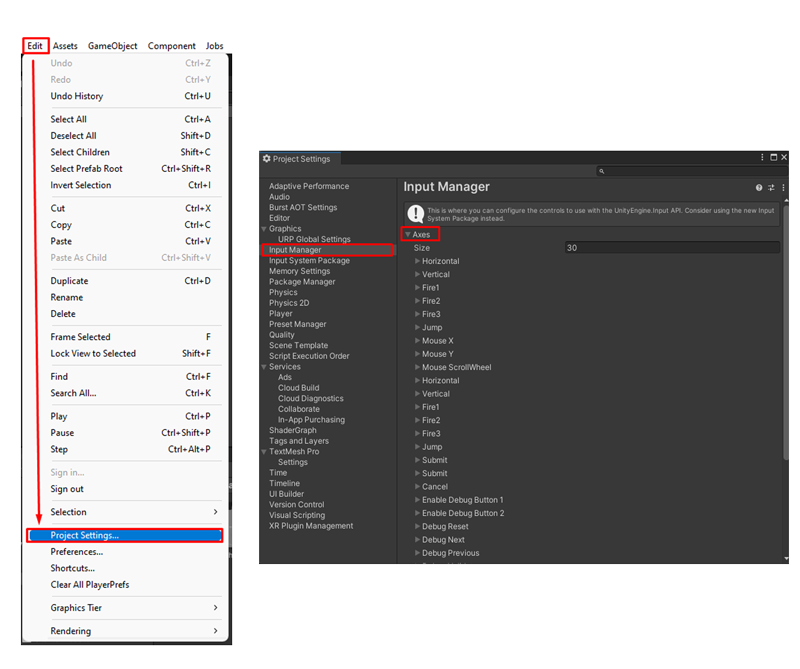


Nota. Tomada de Vegas (s. f.). <http://www.jvunity.weebly.com/uploads/4/7/6/0/47606749/controlmap.png>

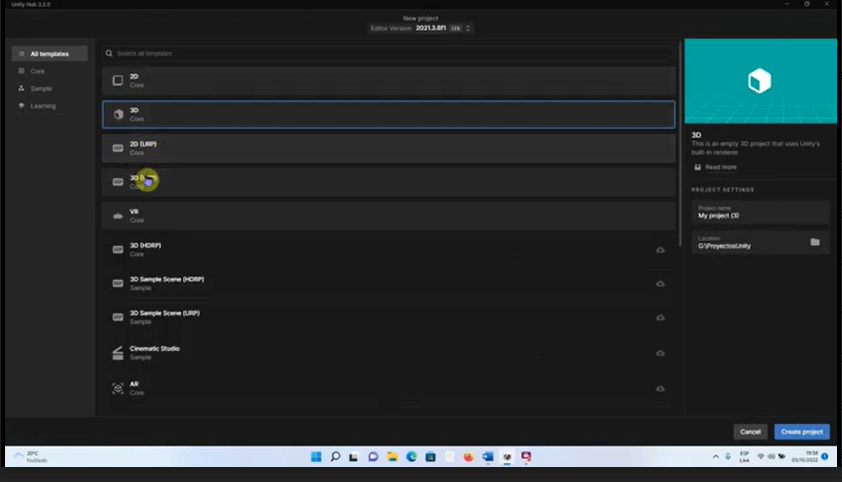
Para acceder a la ventana Input Manager dar clic en el menú Edit > ProjectSettings como se muestra en la siguiente imagen:

**Figura 2**

*Input Manager Unity3D*



En el siguiente video tutorial se observa la programación de personajes controlados por el usuario y la creación de un nuevo proyecto.



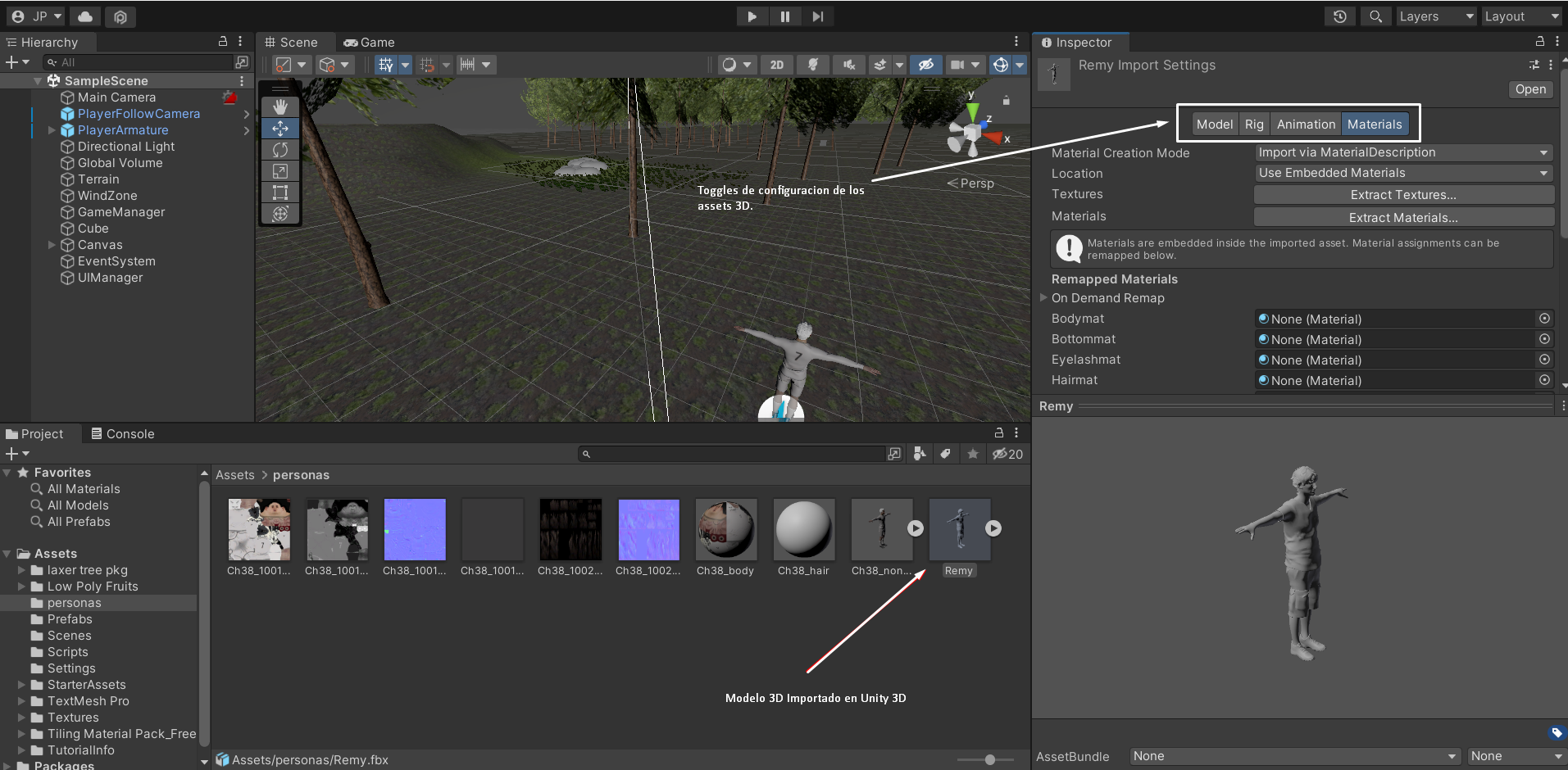
*Input Manager*

1.1. Configuración de animaciones de los personajes

El motor de desarrollo Unity 3D posee diferentes herramientas en su editor que permiten manipular las animaciones de los personajes importados. Las animaciones de los personajes pueden encontrarse embebidas en el personaje o externas al personaje.

**Figura 3**

*Editor Unity 3D*



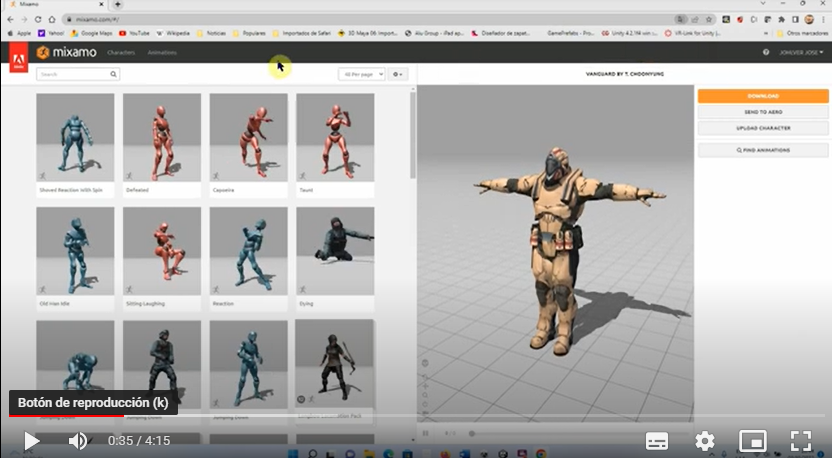
En la web existen un gran número de recursos que facilitan los procesos de animación de personajes si no se tienen las habilidades o no se tienen las animaciones para llevar a cabo este proceso. Para este componente se va a trabajar con la página Mixamo ([www.mixamo.com](http://www.mixamo.com)), que permite realizar las animaciones de personajes de un juego.

Para acceder a la página de Mixamo y comenzar a hacer las animaciones se deben seguir estos pasos.

| Pasos  Ver anexo: CF09\_1.2\_Pasos\_configuracion\_animaciones |
| --- |

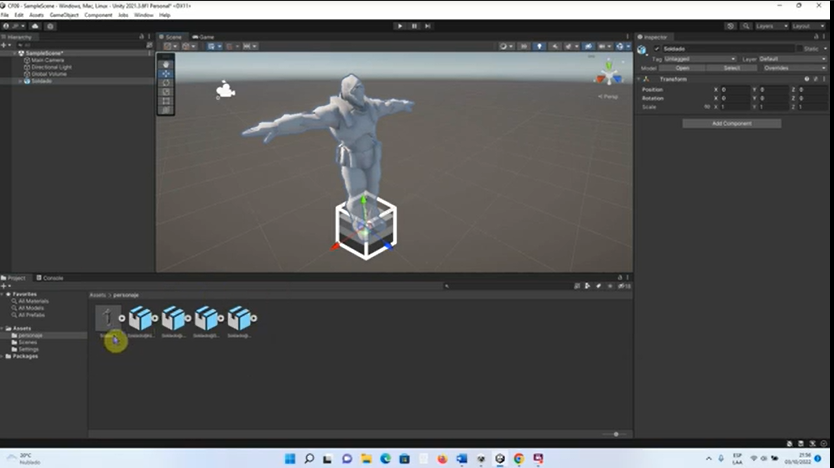
En los siguientes videos tutoriales se explica cómo se pueden generar personajes y darles diferentes animaciones.

* **Obtener modelo 3D en MIxamo**



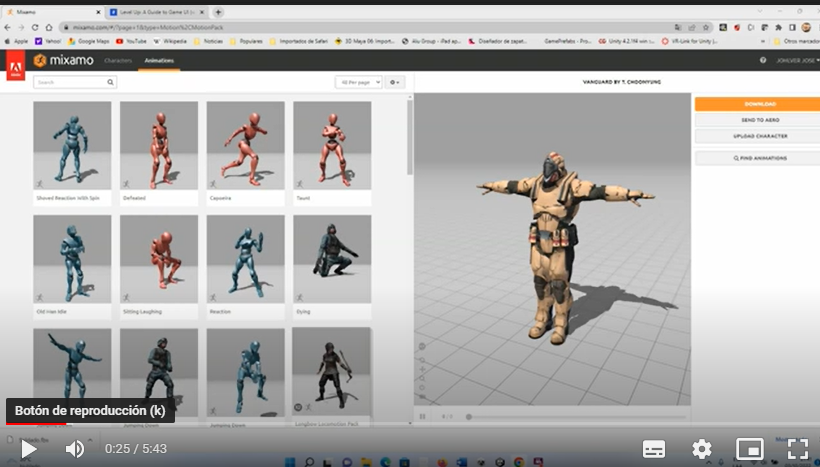
*Obtener modelo 3D en Mixamo*

* **Importar personaje**



*Importar personaje*

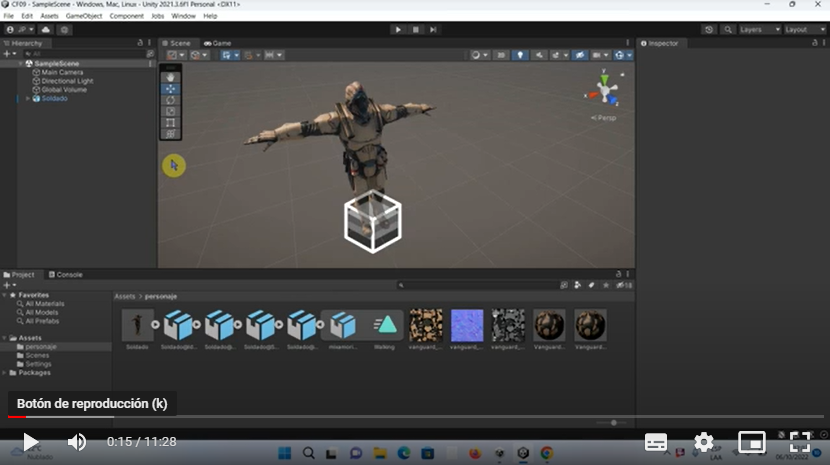
* **Asignar animaciones**

**

*Asignar animaciones*

1.2. **Máquina de estado finito (Animator Controller)**

El Animator Controller es un panel del editor de Unity3D que permite organizar y mantener un conjunto de clips de animación y asociar estas animaciones por medio de transiciones. En la gran mayoría de los casos, el Animator Controller puede contener varias animaciones y cambiar entre cada una de ellas dependiendo de las condiciones que ocurran en el juego, como se puede observar en el siguiente video.



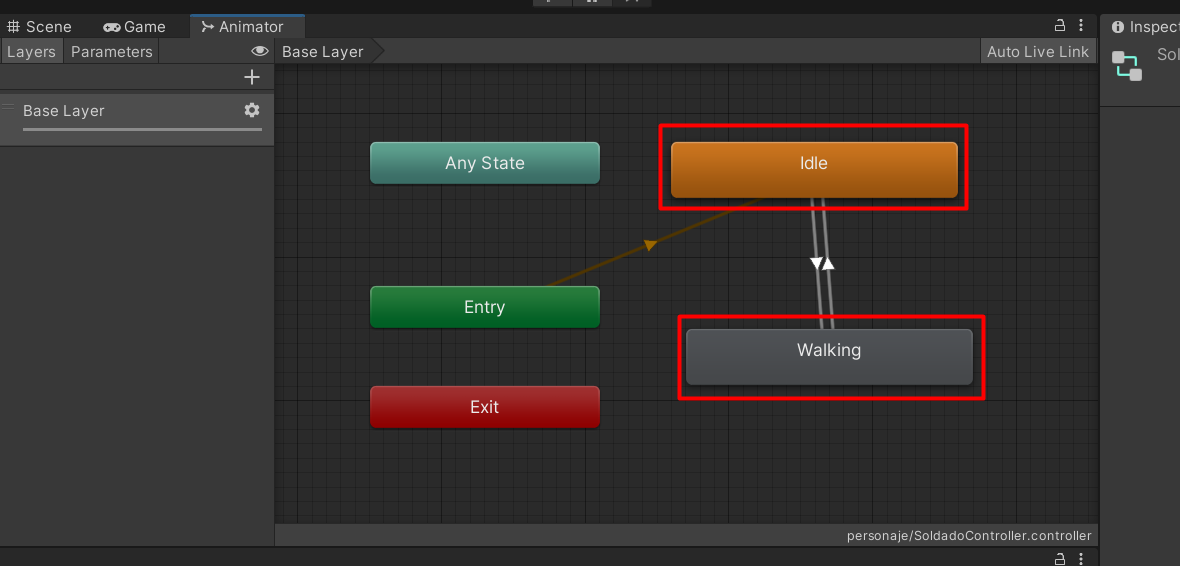
*Animator Controller*

* **Estados de animación**

Los estados en la máquina de estados finito (Animator Controller) son la representación de los comportamientos que el personaje va a tener (reposo, caminar, correr y saltar).

**Figura 4**

*Estados de la FMS*



* **Parámetros**

Los parámetros en la máquina de estados finito representan una variable que permite controlar el cambio de un estado a otro haciendo uso de una transición. Los parámetros cuentan con cuatro tipos:

* Float.
* Int.
* Bool.
* Trigger.

**Figura 5**

*Parámetros FMS*

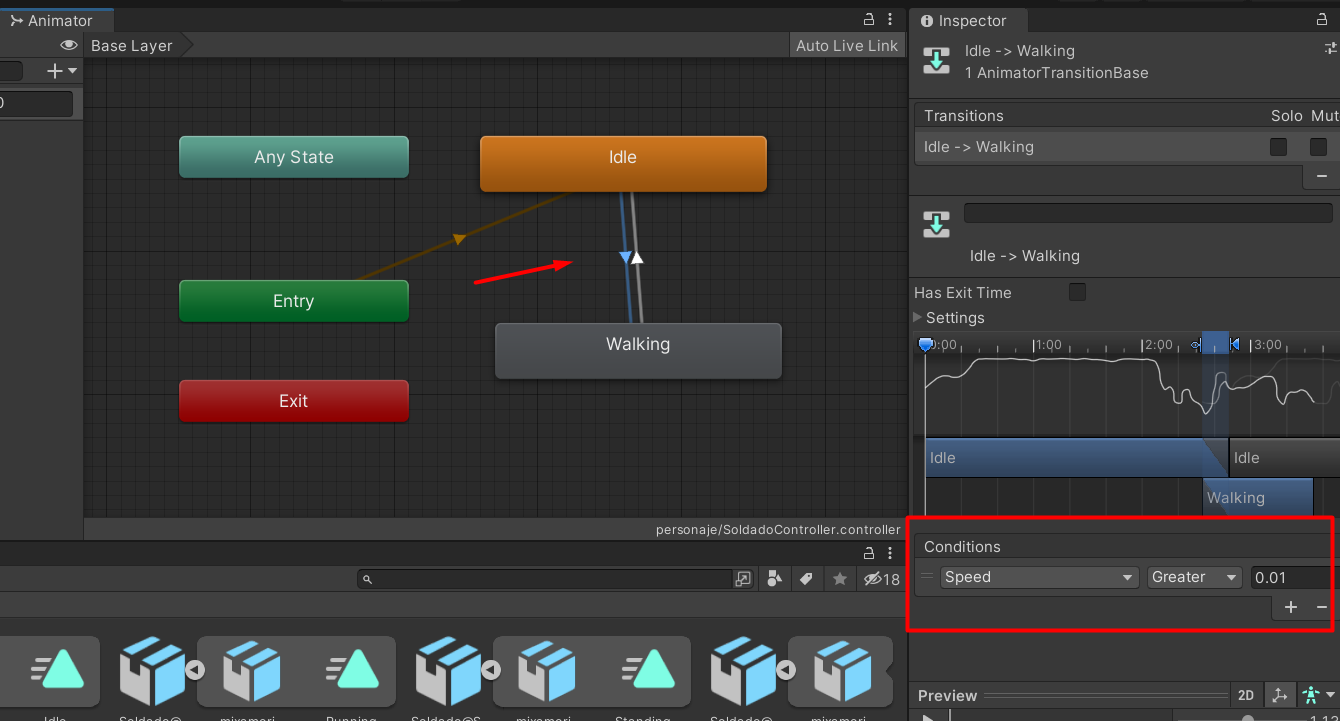


* **Transiciones**

Las *State Machine Transitions* existen para simplificar el proceso de transición entre los diferentes estados del Animator Controller. Las transiciones permiten tener un mayor nivel de complejidad sobre la lógica de la máquina de estados; y las transiciones poseen condiciones que permiten controlar el proceso de cambio de un estado a otro.

**Figura 6**

*Transiciones FMS*

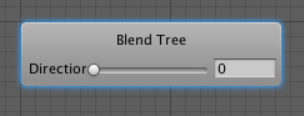


* **Árboles de mezclado (Blend Tree)**

El Blend Tree permite realizar combinaciones entre los diferentes *clips* de animaciones y así generar comportamientos complejos.

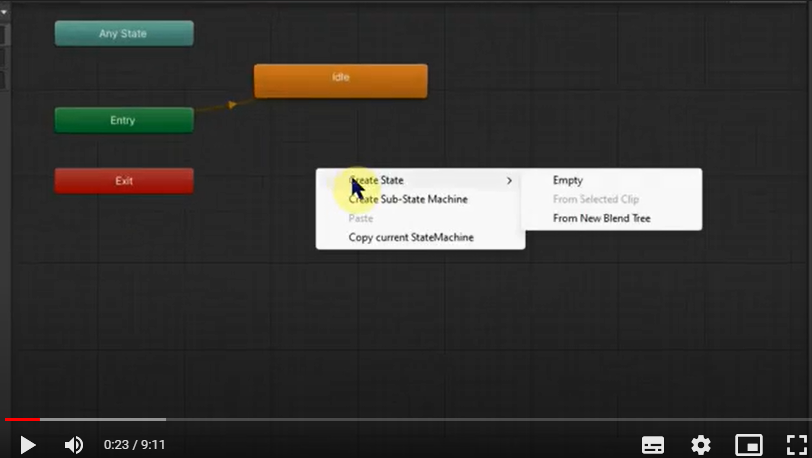
**Figura 7**

*Blend Tree Unity*



blendTree.png

En el video que sigue a continuación se observa cómo se trabaja con el árbol de mezclado o Blend Tree.



*Blend tree*

2. **Programación de eventos de interfaz HUD**

La interfaz de usuario es uno de los principales elementos de los niveles de juego ya que por medio de estos, los usuarios pueden monitorear el comportamiento del nivel de juegos. Por ejemplo, pueden conocer cuánto tiempo les queda de juego, cuánto nivel de salud les queda, etc.

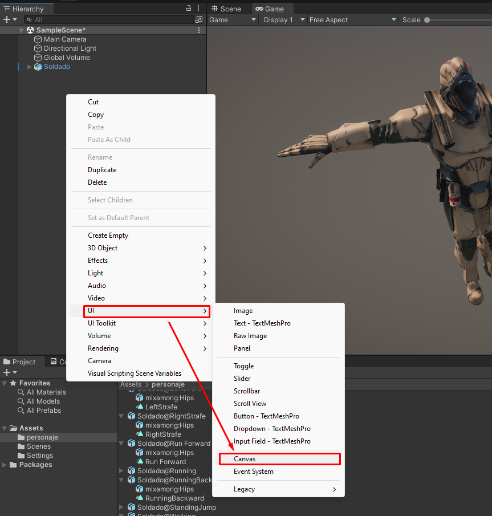
**Canvas**

Unity3D en su conjunto de GameObject posee uno llamado Canvas que es un lienzo que permite organizar de forma detallada la interfaz de usuario. El Canvas trabaja con un componente adicional llamado el EventSystem el cual permite detectar los eventos que se ejecutan cuando el usuario interactúa con los elementos de la UI.

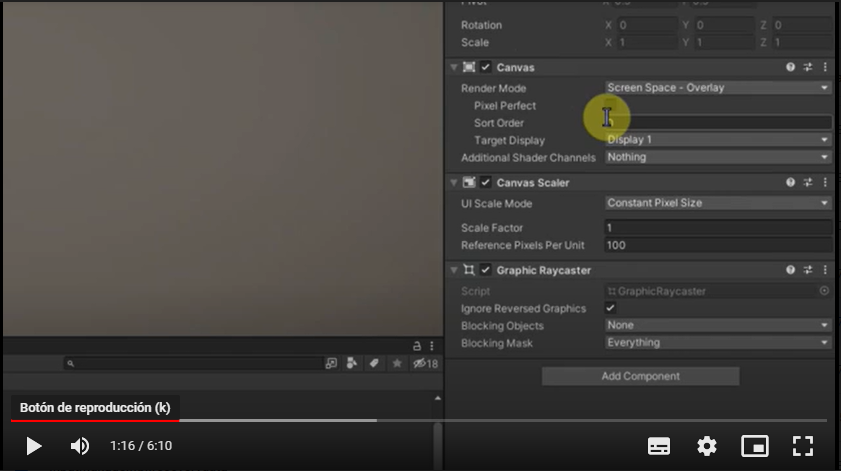
Para agregar el Canvas a la escena hacer clic derecho sobre el panel de jerarquía y seleccionar la opción UI>Canvas.

**Figura 8**

*Canvas*



Para manipular y crear interfaces de usuario con Canvas, se invita a ver este video:



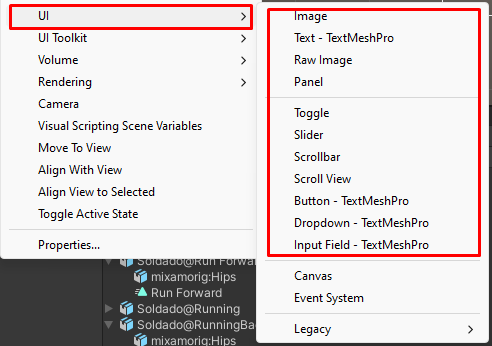
*Iniciando Canvas*

2.1. **Componentes Canvas**

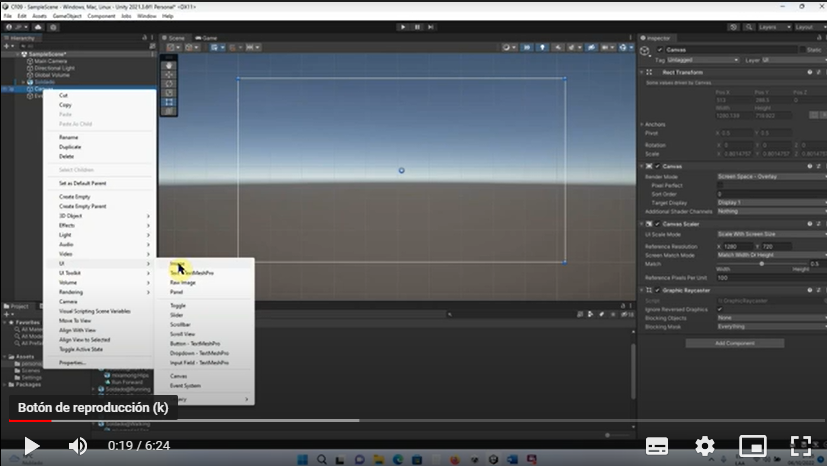
Los componentes del Canvas son los elementos usados para realizar el diseño de la interfaz de usuario. Los principales componentes de la clase UI son los siguientes:

**Figura 9**

*Componentes Canvas*



En el siguiente video se analizan cada uno de los componentes del Canvas.



*Componentes Canvas*

2.2. **Interfaces de usuario responsivas**

Las interfaces responsivas son aquellas que se ajustan de forma automática a cualquier resolución de pantalla. Unity3D tiene una serie de elementos que permiten realizar este tipo de configuraciones en las interfaces de usuario, a saber:

| Slide Simple  Ver anexo: CF09\_2.2\_Slide\_Elementos\_RecTransformer |
| --- |

En este video se observa cómo funciona el componente Rec transformer para manipular cada uno de los elementos del Canvas.



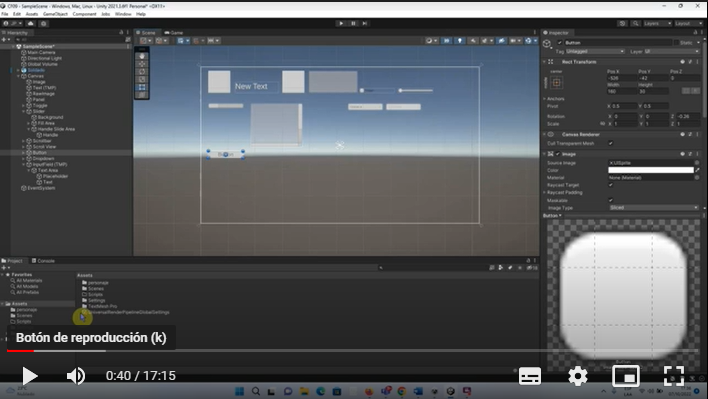
*Rec transformer en Canvas*

2.3. **Eventos**

Los eventos en la programación son acciones que se ejecutan cuando los usuarios interactúan con los diferentes elementos de la interfaz de usuario. En la siguiente tabla se describe cada uno de esos elementos.

| Evento | Elemento UI | Descripción |
| --- | --- | --- |
| OnClic | Button | Permite ejecutar una acción cuando el usuario interactúa con algún botón de la UI. |
| OnValueChange | DropDown | Ejecuta una acción cuando el jugador selecciona un elemento de la lista desplegable. |
| OnValueChange | InputField | Ejecuta una acción cuando cambia el valor de la caja de texto. |
| OnEndEdit | Ejecuta una acción cuan se finaliza el proceso de edición del texto del elemento. |
| OnSelect | Ejecuta una acción cuando se selecciona la caja de texto. |
| OnDeselect | Ejecuta una acción cuando de deselecciona la caja de texto. |
| OnValueChanged | Slider | Ejecuta una acción cuando el valor del slider ha cambiado. |
| OnChangeValue | Toggle | Ejecuta una acción cuando el valor del elemento ha cambiado. Retorna un valor booleano (*true* o *false*) |

En el siguiente video se explica cada uno de los elementos de la interfaz de usuario, que componen la UI.



*Eventos Interfaz de Usuario*

Para ampliar el tema sobre la programación de eventos de interfaz HUD, se sugiere consultar en el material complementario la página Unity Learn el tema de IU Components.

3. **Programación de eventos disparadores (interacción con el entorno)**

En el diseño de las mecánicas del videojuego es necesario crear interacciones con el entorno, es en este punto donde los eventos ocurridos por condiciones físicas permiten a los jugadores recoger ítems de los niveles de juego, sufrir daño por pasar sobre zonas no permitidas, etc.

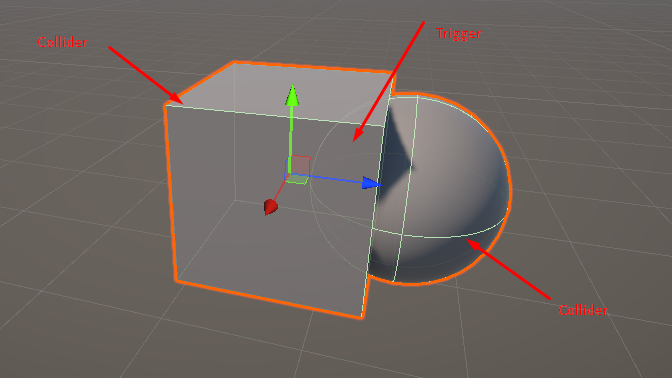
Los motores de videojuegos permiten simular los comportamientos físicos del entorno real haciendo uso de diferentes componentes como *rigibodies*, materiales físicos, colisionadores etc., permitiendo que las mecánicas sean dinámicas y parecidas a la realidad.

***Triggers***

Los *triggers* son eventos que ocurren cuando los objetos de videojuego se entrelazan entre sí como se observa en la figura.

**Figura 10**

*Trigger entre gameobjects*

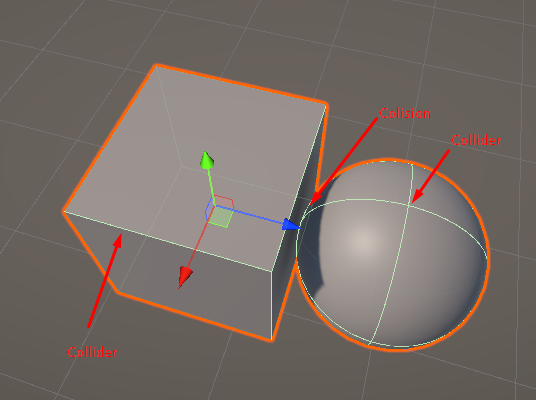


***Collision***

La colisión es un evento que ocurre cuando dos *gameobjects* se tocan entre sí, este evento es muy utilizado para provocar daño a los enemigos en un videojuego.

**Figura 11**

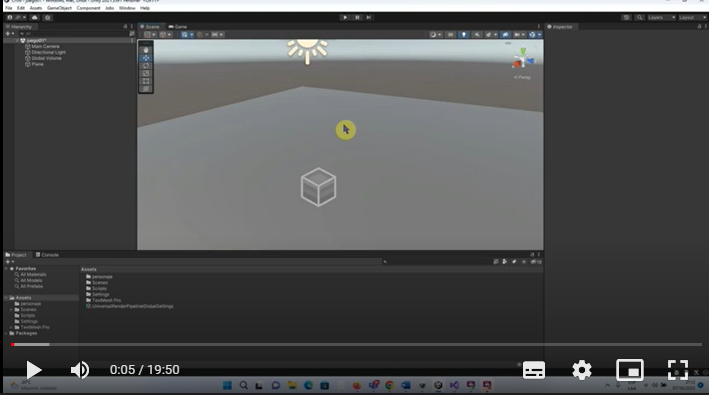
*Collision*



Los tipos de *colliders* son:

| Sphere Collider  Este tipo de *collider* es muy usado en formas redondas. | Box Collider  Es empleado en formas cubicas, edificios, carretera, etc. | Capsule Collider  Utilizado en los personajes para crear zonas de daño; es muy utilizado en los troncos de los árboles. |
| --- | --- | --- |

En el siguiente video se estudian las físicas y colisiones en el motor de desarrollo.



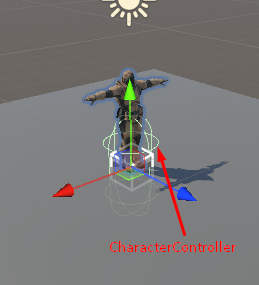
*Físicas y colisiones*

**OnControllerHit**

Este tipo de colisión es característico en los personajes de los videojuegos, ya que permite detectar cuando el jugador recoge elementos del nivel de juego.

**Figura 12**

*Character controller*



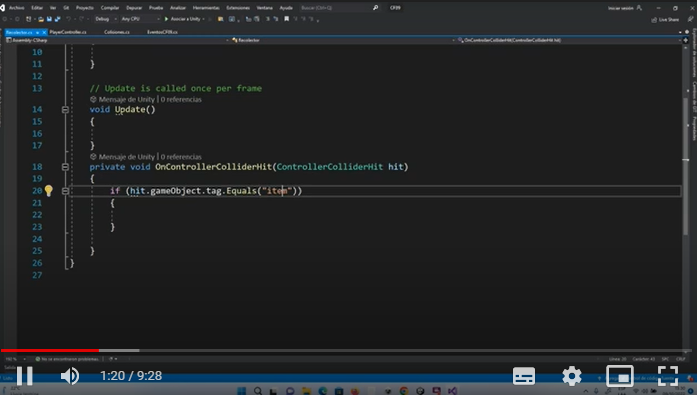
A través de los siguientes videos se explicará cómo controlar un personaje y aplicar animaciones:

* Character controller para controlar el movimiento de un personaje.



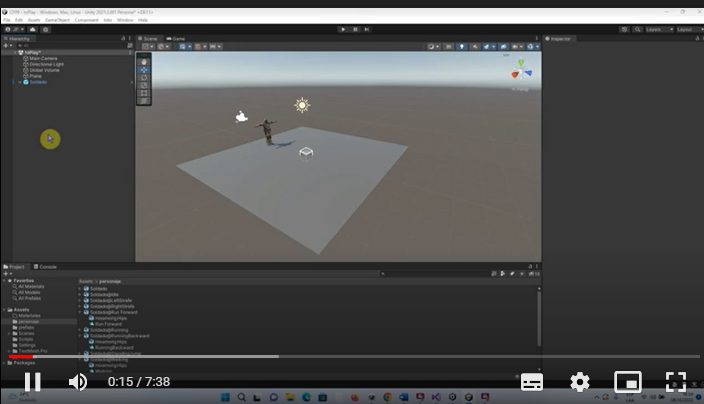
*Character Controller*

* Colisión con otros objetos de la escena.



*Collider Hit*

* Generación de cámara de seguimiento al personaje.



Cinemachine

Para ampliar el tema sobre la programación de eventos disparadores e interacción con el entorno, consultar en el material complementario la página Unity Learn el tema de Catapult Physics; Forces, and Energy.

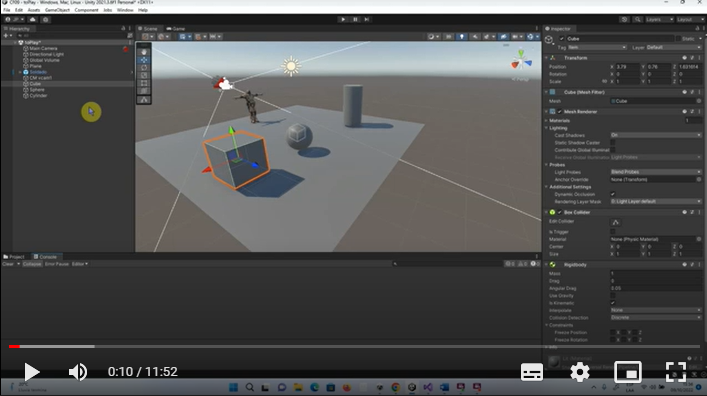
**4. Programación de comportamientos relativos a la mecánica de videojuego**

La interacción del jugador con los objetos del videojuego desempeña un papel muy importante en las mecánicas del videojuego, ya que así puede saber el estado actual del nivel, los elementos que ha recolectado para su inventario, qué nivel de vida le queda, entre otros muchos aspectos más.

**Interactividad UI Recolección de ítems de juego**

Una de las principales actividades que se deben cumplir en la gran mayoría de mecánicas es la recolección de objetos en el escenario; esto hace que el jugador pueda recoger pociones mágicas, pociones de salud e incluso municiones para sus armas.

Continuando con las colisiones del Character Controller, en el siguiente video se explica la creación de un sistema de recolección y conteo que se visualiza en la interfaz de usuario (UI).



*Recolector*

5. **Programación de inteligencia artificial**

Los desarrolladores de videojuegos viven preocupados por ofrecer juegos que contengan las mejores experiencias gráficas y una mayor inmersión en cada mecánica desarrollada. La inteligencia artificial en los videojuegos también juega un papel fundamental ya que permite tener grandes comportamientos de los jugadores NPC, haciendo que cada nivel de juego sea más atractivo.

Unity 3D, al igual que muchos motores de desarrollo de videojuegos, posee una estructura bastante robusta que permite integrar de forma sencilla la inteligencia artificial a nuestros juegos. Para tener una visión general de la navegación en Unity, son necesarios los siguientes elementos:

| Slide Simple  Ver anexo: CF09\_5\_Slide\_Navegacion\_Unity |
| --- |

**Agregar agente de navegación**

Estos son los pasos para agregar el agente de navegación:

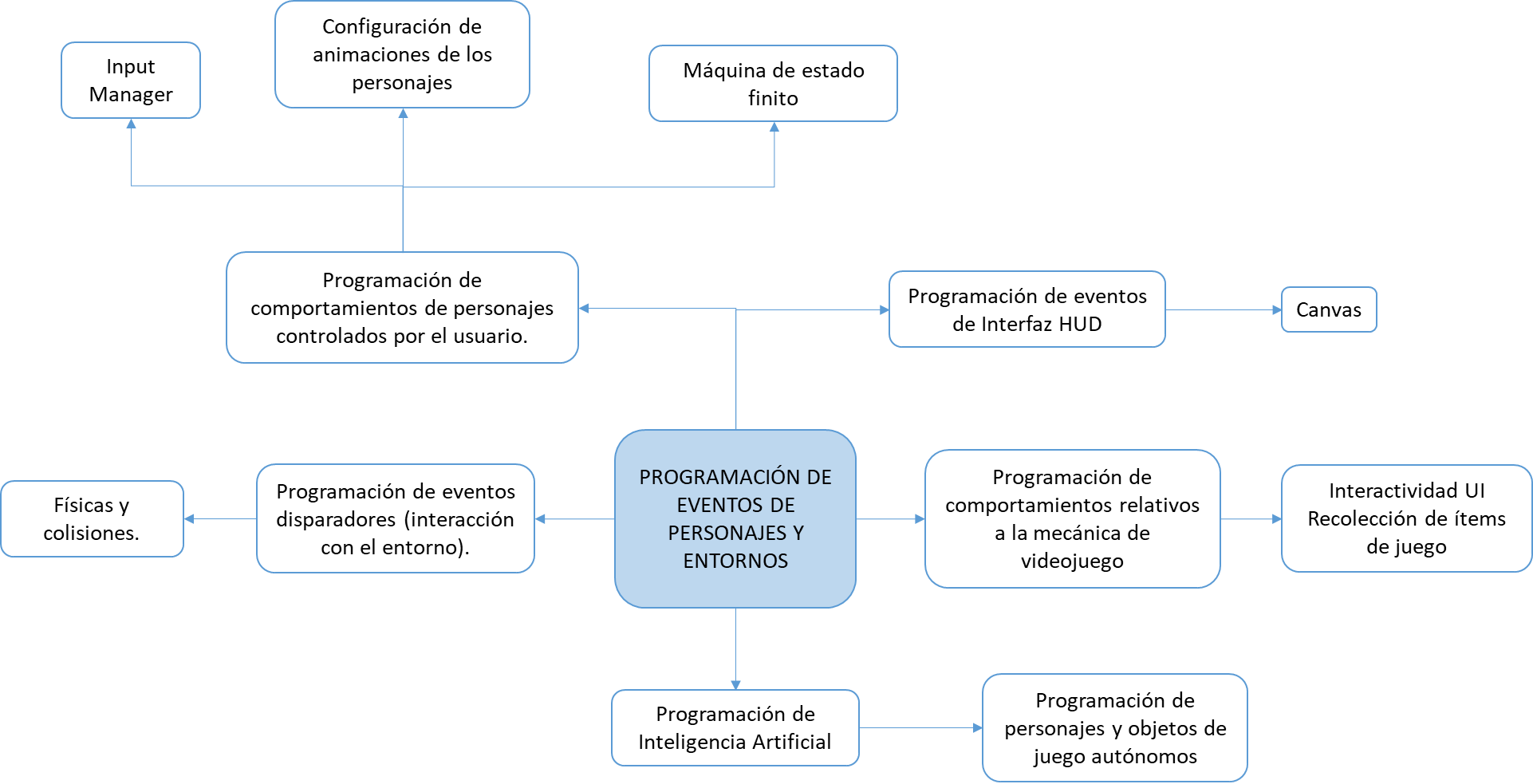
| Pasos  Ver anexo: CF09\_5\_Pasos\_agregar\_agente\_navegacion |
| --- |

Para ampliar el tema sobre Programación de Inteligencia Artificial, consultar en el material complementario la página Unity Learn, el tema de Navigation Meshes.

1. **Síntesis**

En el siguiente esquema se presenta una recopilación de los temas trabajados en este componente formativo **“Programación de eventos de personajes y entornos”.**

**Figura 13**

*Síntesis programación de eventos de personajes y entornos*

1. **ACTIVIDADES DIDÁCTICAS (Se debe incorporar mínimo 1, máximo 2)**

| SCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| --- | --- |
| Nombre de la Actividad | Actividad 1: Componentes interfaces de usuario  Actividad 2: Conocimiento en videojuegos |
| Objetivo de la actividad | Actividad 1: Identificar los diferentes componentes usados en la programación de eventos y personajes en los videojuegos.  Actividad 2: Identificar los eventos de los componentes usados en el diseño de interfaces de usuario. |
| Tipo de actividad sugerida | Act 1.Act2. Relacionar términos |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | Anexos*:*  CF09\_228108\_Actividad\_didactica\_1  CF09\_228108\_Actividad\_didactica\_2 |

1. **MATERIAL COMPLEMENTARIO**

| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| --- | --- | --- | --- |
| 1. Programación de eventos de Interfaz HUD | Unity Technologies. (2022). *IU Components.* Unity Learn. <https://learn.unity.com/tutorial/ui-components> | Website | <https://learn.unity.com/tutorial/ui-components> |
| 1. Programación de eventos disparadores (interacción con el entorno) | Unity Technologies. (2022) *Catapult Physics; Forces, and Energy.* Unity Learn. <https://learn.unity.com/project/catapult-physics-forces-and-energy> | Website | <https://learn.unity.com/project/catapult-physics-forces-and-energy> |
| 1. Programación de Inteligencia Artificial | De Byl, P. (2022). *Navigation Meshes* Unity Learn. <https://learn.unity.com/project/navigation-meshes> | Website | <https://learn.unity.com/project/navigation-meshes> |

1. **GLOSARIO:**

| TÉRMINO | SIGNIFICADO |
| --- | --- |
| FMS | acrónimo usado en Unity para referenciar la máquina de estado finito. |
| *Input* | clase de Unity usado para detectar cuando el jugador interactúa con un dispositivo de entrada (teclado, *mouse, joystick*). |
| Parámetro | variable definida en una máquina de estado para controlar las transiciones entre los estados. |
| *Public* | modificador de acceso que permite dar accesibilidad a las variables y métodos en un programa. |
| Transición | relación que existe entre cada estado en una máquina de estados finitos. |
| *Using* | palabra reservada en .net que permite importar referencias a clases. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Acerenza, N., Coopes, A., Mesa, G., Viera, A., Fernández, E., Laurenzo, T., y Vallespisr, D. (2009). Una metodología para desarrollo de videojuegos*. 38º JAIIO - Simposio Argentino de Ingeniería de Software*, pp. 171-176. <https://www.fing.edu.uy/sites/default/files/biblio/22811/asse_2009_16.pdf>

Colombiaaprende.edu.co (s. f). *Lección 1: ¿Qué es un algoritmo?.* <https://aprende.colombiaaprende.edu.co/sites/default/files/naspublic/curriculos_ex/n1g10_fproy/nivel1/programacion/unidad1/leccion1.html>

GameDevTraum. (2021). *Scripts en programación.* <https://gamedevtraum.com/es/programacion-informatica/introduccion-a-la-programacion/que-es-script-programacion/>

Lenguajes de programador.net. (s. f). *¿Qué es una variable en programación?.* <https://lenguajesdeprogramacion.net/diccionario/que-es-una-variable-en-programacion/>

Lenguajes de programador.net. (s. f). *¿Qué es una función (o un método) en programación?.* <https://lenguajesdeprogramacion.net/diccionario/que-es-una-funcion-o-metodo-en-programacion/>

Unity Technologies. (2022) *Unity Learn*. <https://learn.unity.com/>

Unity Technologies. (2016). *Unity Documentation.* Manual Unity. <https://docs.unity3d.com/es/530/Manual/UnityManual.html>

1. **CONTROL DEL DOCUMENTO**

|  | Nombre | Cargo | Dependencia  *(Para el SENA indicar Regional y Centro de Formación)* | Fecha |
| --- | --- | --- | --- | --- |
| Autor (es) | Johlver José Pardo García | Experto Temático | Regional Santander, Centro Industrial de Mantenimiento Integral | Octubre 2022 |
| Luz Aída Quintero Velásquez | Diseñadora Instruccional | Regional Distrito Capital, Centro de Gestión Industrial | Octubre 2022 |
| Rafael Neftalí Lizcano Reyes | Responsable Equipo de Desarrollo Curricular | Regional Santander, Centro industrial del diseño y la manufactura | octubre 2022 |
| Silvia Milena Sequeda Cárdenas | Asesora metodológica y pedagógica | Regional Distrito Capital - Centro de Diseño y Metrología | Octubre de 2022 |
|  | José Gabriel Ortiz Abella | Corrector de estilo | Regional Distrito Capital - Centro de Diseño y Metrología. | Octubre del 2022. |

1. **CONTROL DE CAMBIOS**

**(Diligenciar únicamente si realiza ajustes a la Unidad Temática)**

|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| --- | --- | --- | --- | --- | --- |
| Autor (es) |  |  |  |  |  |