Biblioteca > Añadir al proyecto.

Y seleccionamos el proyecto que sea.

**Ensamblador**: lenguaje usado en microprocesadores.

**C**: lenguaje estructurado (60 / 90)

**C++**: lenguaje orientado a objetos. Capacidad de hacer programas más grandes

**Blueprints**: las cajas se ejecutan de izquierda a derecha y siguiendo los cables.

Habrá un puerto/cable blanco. Y otros puertos

* A la izq son inputs (parametros).
* A la derecha outputs

Por ejemplo un print, el input es el string y el output no tiene.

El **CÓDIGO** de las blueprints viene siempre con **EVENTOS**.

OnBegin podría ser

* Del nivel
* De un objeto

## Hola Mundo

Para hacer un hola mundo, onBegin del nivel.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Open Level Blueprint: editor de Blueprints.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamenteCaja roja: evento. Tienen un puerto blanco de salida y no tiene ninguna deentrada.

* Rosa: String
* Rojo: Bool
* Azul: Vector3
* Verde: double

Al salirnos podemos darle a compilar, aunque creo que compila solo.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Es Multithreading: en el delay hay una espera. Entra en Print String y Delay a la vez. En el Print del final entra a los 3 segundos.

Pantalla de un video juego

Descripción generada automáticamente con confianza media

**DEBUGGER**: Podemos poner BreakPoints e ir paso a paso (add, remove, disable breakpoint)

Interfaz de usuario gráfica

Descripción generada automáticamente

F10 para ir paso a paso línea por línea

Puedo mirar en los puertos izquierdos para ver el estado en cada momento.

## Objetos con lógica

* **Actores**: tienen varias piezas
  + Visuales (plataforma por ejemplo, lo que se ve)
  + Lógica
* **Pawns**: objeto con IA
* **Character**: es un actor que puede ser controlado por el player

**Vamos a hacer una plataforma destruible**

En el content Drawe, click derecho Blue Print ClassD

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Vamos a crear un actor.

Ahora vemos esto: Cuando nos metemos en algo del nivel como antes, no hay geometría, solo está el Event Graph. Ahora tenemos 3 pestañas:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

En los components le damos a ADD y podemos añadir cosas (COMPONENTES).

Destroy actor. El target es self (this).

Aunque esto no se hace normalmente así, se le añade un trigger encima de la plataforma, cuando entremos dentro entonces que se muera la plataforma.

Le estoy poniendo un evento al componente. No es lo mismo que si se lo añado a la Blueprint, antes ha intentado ponerle el overlap a la blueprint y no ha funbcionado.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Vamos a usar ahora el Tick

Pantalla de juego de computadora

Descripción generada automáticamente con confianza baja

En los Details del Collider podemos poner el HiddenInGame para que podamos verlos en el juego.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Cylinder platform y creamos 2 spehere collisiones

Pantalla de computadora con fondo negro

Descripción generada automáticamente con confianza media

La posición realtiva se consigue restando dos absolutas.

Pantalla de juego de computadora

Descripción generada automáticamente con confianza baja

Coger una construcción del Mario chorra y la hacemos. Por ejemplo:

Blueprint con dos componentes (switch y puerta).

Hay un pulsador que cuando me pongo encima, abre una puerta durante 5 segundos, Y cuando pasen vuelve la puerta.

Se puede hacer que suba la puerta y baje (es lo mejor) o hacer visibility a true o false.

Comunicaciones entre objetos. Yo cojo monedas y que haya un contador al que le pasan mensajes al contador. Y también IA