# Compilar automaticamente:

Editor Preferences

A screen shot of a phone

Description automatically generated with medium confidence

# Logs

Podemos ponerlos como con capas de prioridad. El log del componente Interact a lo mejor ahora no lo necesito, pues le bajo la prioridad (o desactivo el log).

Tenemos que sentirnos cómodos con esto.

## Ejemplo

In .h of MCVInteractInterface.h

DEFINE\_LOG\_CATEGORY\_EXTERN(LogInteract, Log, All) (declare NO)



* 2º param: Log / Display / Verbose / VeryVerbose: si pongo Log se muestra todo lo del nivel Log y lo superior.
* 3er param: lo que se compila. (All, se compila todo; Warning, por debajo de log no se compila).

En el .cpp



When I put UE\_LOG(LogTemp, Display …) now I change it:





## Tipos por prioridad

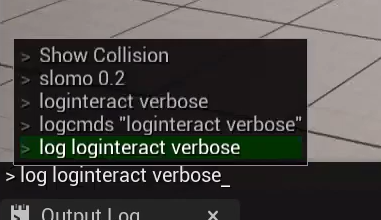
* **Display**: para mostrar algo importante
* **Verbose**: no tan importante
* **VeryVerbose**: algo que no hace falta verlo de normal
* **Warning**: aviso, algo podría fallar.
* **Error**: algo que no debería pasar.
* (etc… se pueden ver en el código)

### Cambiar la prioridad en ejecución

There is a Console where we can type: **log loginteract verbose**. (Presionar Tecla º)

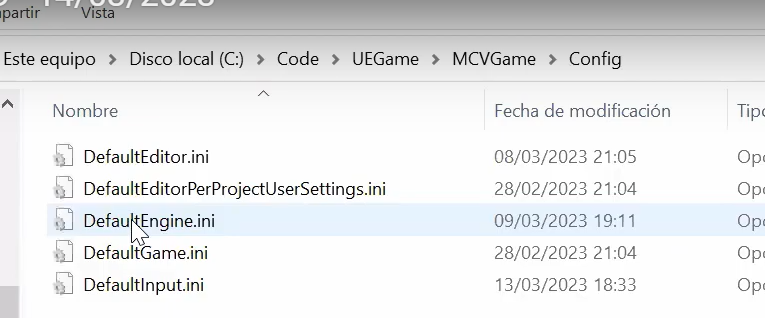
**Graphical user interface

Description automatically generated with medium confidence**

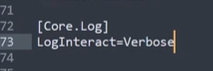
****

### Cambiar prioridad en DefaultEngine.ini

En el DefaulEngine.ini

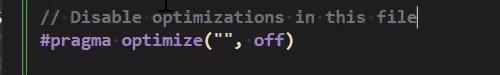
****

If we type this at the end, it overrides what there is in the .cpp

****

(el ini se puede cambiar sin tener que compilar)

This doesn’t optimize from this line. If a variable is not used for example it doesn’t optimize it



At the end of the file we have to active it again.



We can do it involving only a function.

## Data tables

Yo tengo mis items del inventario y cada uno tiene un nombre. Ese item luego tendrá una malla, unos datos… etc.

Me serviría un diccionario para que cada itemID tenga una serie de propiedades (icon, tags, mesh…)

Esos objetos no existen en C++.

Creamos para ello un DATA\_TABLE

Usan un ID (FName) como ID.

Se usa para definir tipos de enemigos, ítems, …

Esa tabla la puedo exportar a un csv (fuera) y luego la puedo reimportar.

**Necesitamos la datatable en todo el juego. Lo ponemos en la GameInstance.**

**Define el tipo de Row:**

Text

Description automatically generated

BlueprintType para que se vea desde BP. El struct empieza por F porq no es UObject

Crear un DataTable a partir del TypeRow definido anteriormente

Graphical user interface, application

Description automatically generated

Y le podemos poner un tipo

Vamos al GameInstance

A black screen with white text

Description automatically generated with low confidence

Desde BP nos sale esto:

Graphical user interface, application

Description automatically generated

Y seleccionamos la de ítems

EL objetivo es que pasemos del inventario este de “Water” a que ponga agua y un icono.

A picture containing electronics, keyboard

Description automatically generated

Hay que acceder al diccionario (datatable).

Y le ponemos al UIInventory esto que será la traducción:

Text

Description automatically generated

Text

Description automatically generated

Y en el SET DATA del WBP\_InventoryItem:

A screenshot of a video game

Description automatically generated with medium confidence

Ahora vamos a asociar una mesh a cada item y un color

En el MCVItemTypes

Graphical user interface, text

Description automatically generated with medium confidence

HardReference. Mi data table tiene asociado este objeto. Al cargar la DT, cargo tamb el objeto.

Entonces vamos a usar una SOFT REFERENCE.

Esto lo podemos ver en la Reference Viewer

A screenshot of a computer

Description automatically generated with medium confidence

Blanco: hard reference ¿?

Rosa: soft reference

Probamos con Soft también

Text

Description automatically generated

A screenshot of a video game

Description automatically generated

Para estas soft textures se puede llamar a async load asset

Graphical user interface, application, Teams

Description automatically generated

Para que cuando cargue entonces lo muestro

Deberes: pasar e C++ todo lo que ha hecho con Blueprint de crear unn spawn actor cuando se presiona la K, cambiar la static mesh

A screenshot of a video game

Description automatically generated

Ç

Ptr a clase que deriva de actor

Graphical user interface

Description automatically generated

Aquí va lo del BP\_Chest.