En este array de bytes guardamos lo que queramos para cada actor. La idea es que sea genérico.

Text

Description automatically generated

Otra opción es guardar algo así, lo malo es que tendríamos un montón de structs para cada clase.

Graphical user interface, text, application

Description automatically generated

Además de los datos serializados podemos guardar más cosas.

* Si no puedo guardar la transform pues me la puedo guardar tal cual.
* Interesa tamb saber si está destroy

Text

Description automatically generatedText

Description automatically generated

Y en el saver

Text

Description automatically generated

Creamos dos funciones en el UMCV\_SaveGame

* Void SaveActorState (AActor\* ActorToSave)
* Void LoadActorState (AActor\* ActorToLoad)

Hay que serializar las properties de cada actor y guardarlas array de bytes y en el mapa

Text

Description automatically generated

Los actores tienen un método Serialize. Se guarda en un archive. Guarda en binario los datos de un actor, puede ser un json, un binario…

Graphical user interface, application, website

Description automatically generated

Nosotros vamos a escribir en memoria (en el mapa)

Nos va a devolver siempre una referencia, si no existe la crea:



Necesitamos un archive para escribir. No podemos usar directamente:



Entonces así:

Text

Description automatically generated

Pero para recuperar luego perderíamos info, entonces se usa un proxy: los FNames y los objetos los guarda como un string en el archive que hay por debajo, porq si quiere guardar un puntero lo guardaría tal cual, a saco, entonces no nos serviría. El proxy traduce los ptr a objects como strings:

Text

Description automatically generated

Finalmente:

Text

Description automatically generated

Crea otra función:



Text

Description automatically generated

Y a la otras le cambia nombre:

Text

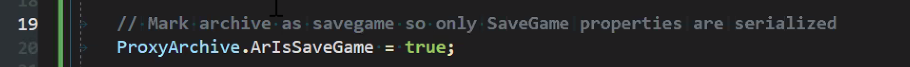
Description automatically generated

Y en el ToggleDoor, cada vez que se acbre o ierra la puerta:

Text

Description automatically generated

Añade un flag en el SaveActorRecord en ProxyArchive.ArIsSaveGame



Porq si no serializaría todas las UPROPERTY del actor y eso es una burrada.

Abre la puerta y comprueba que se guardan los datos con un breakpoint y mirando dentro de la variable.

Vale ahora queremos cargar la información cuando arranca el juego (puede ser beginPlay) o incluso antes: PreInitializeComponents (lo pone en el MCV\_Door)

Text

Description automatically generated

En el mapa tendremos información (con nombres de actores y su info) necesitamos recuperarla y pasárselo al actor

Ahora no queremos un findOrAdd, solo un find para ver si existe (antes si no existía lo creaba para guardar los datos, ahora solo necesito recuperarlos).

Text

Description automatically generated

Al iniciar el juego d enuevo cambia el estado de la puerta: el problema es que cambia el estado pero no llama a la animación de la puerta para cerrarse.

Lo que hace es devolver un bool, y llamar a RequestTick() creo que een el preinitializeComponents

FUNCIONA

Queremos poder guardar más información, Por ejemplo con algo que se pueda destruir. Vamos a crear un botón para destuir la puerta.

Text

Description automatically generated

Y en el LoadActor al princpio comprobamos si está DESTROYED

Text

Description automatically generated

Hecho lo de destroyed bien PERO

Text

Description automatically generated

Tenemos que poner esto en cada Actor. Entonces queremos encapsular esto en un componente. De manera que yo le pongo el componente y el solo ya verá si está marcado como destruido pues que se borre.

MCVGame/Compnents/MCVSaveGameComponent