# Intro

* Acabar con el gamemode
* Player State
* Hacer CompLife
* Behaviour Tree si da tiempo

# Gamemode

## Timer

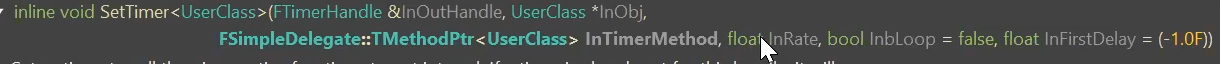
We create a Timer. It needs a Handle



Create a function OnCinematicTimer in GameGameMode

We SetTimer in HandleStartingNewPlayer\_Implementation

It needs the following parameters:

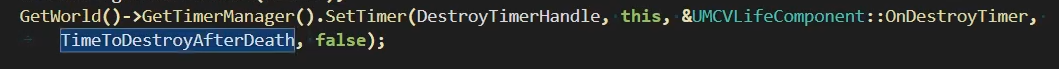


* FTimerHandle: handle we have already defined
* UserClass: **this** in this case (GameGameMode)
* FSimpleDelegate: function of ObjClass
* float InRate: how much time it lasts to start the timer
* InbLoop: is this loopable
* Float inFirstDelay: a first delay

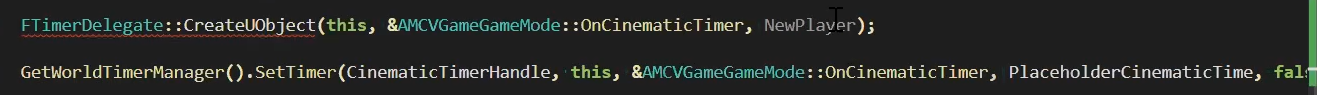
So we call the Timer like this:



(If we are inside a Component MCVLifeComponent, we can’t access to the getWorldTImerManager so we do this like that). We need to create the FTimerHandle as well



Function before BeginPlay 🡪 InitializeComponent

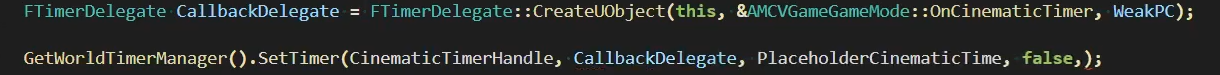
Now, we need to pass as a parameter of the OnCinematicTimer a PlayerController. To do this we can: 

And the function now receives the PlayerControllerText

Description automatically generated

However it is a bad idea to store a RawPtr to a Class because it can be null when we call the function. Solutions:

* Store it as a Member of a Class and use the UPROPERTY() attribute
  + It stores a Strong Reference
* TWeakObjectPtr<APlayerController>: now if the ptr is deleted
  + Weak Reference



And in the funct OnCinematicTImer we can get the Raw Ptr with Get() and if NewPlayer.IsValid

Text

Description automatically generated

Now we are spawning the player after X seconds.

# PlayerState

Create a class MCVGamePlayerState

Create a USTRUCT for Player Progression (XP and level)

Text

Description automatically generated

Use delegates when level up. So the Life or UI can listen to this.

**Multicast Delegate** (2 params: levelPrevious, newLevel)

DECLARE\_MULTICAST\_DELEGATE\_TwoParams(FOnPlayerLevelUpDelegate, int, int);

We are defining the type of the delegate, so we create a variable Delegate



We create a function to AddXP and use the delegate

Text

Description automatically generated

We call the function with Broadcast.

Now we need to bind (or subscribe) to the delegate.

In **MCVGameCharacter.cpp** 🡪 #include “GameFramework/MCVGamePlayerState.h”

There is a function PossessedBy that we can override and it is automatically called when changes the possession

Graphical user interface, website

Description automatically generated

And we add the function. We can use:

* AddUObject
* AddLamba
* AddSP (shared pointer)

Text

Description automatically generated

And we add XP when interacting, for example

Text

Description automatically generated

# Macros para delegates

DECLARE\_MULTICAST\_DELEGATE\_TwoParams(FOnPlayerLevelUpDelegate, int, int);

Se pueden registrar muchas funciones

DECLARE\_MULTICAST\_DELEGATE\_TwoParams(FOnPlayerLevelUpDelegate, int, int);

Solo se puede registrar una función (no es lo normal)

DECLARE\_DYNAMIC\_MULTICAST\_DELEGATE\_TwoParams(FOnPlayerLevelUpDelegateDynamic, int, PrevLevel, int, CurrentLevel);

Para registrar desde Blueprints se usan los DYNAMIC\_MULTICAST

Necesitan el nombre de la variable tamb.

Declare variable

Text

Description automatically generated

Call Broadcast

Text

Description automatically generated

To listen to the Dynamic Delegate is a little bit different:



NOTE: we need to mark the OnLevelUp function as UFUNCTION()

We can unbind too, so we store a FDelegateHandle

Graphical user interface

Description automatically generated with low confidence

And we want to unbind when unpossessed so override the function



We store the Handle when we add the function



For Dynamics:



And we remove (unbind) like thi s

Text

Description automatically generated

# Comp Life

Create a compLife.h

Text

Description automatically generated

Text

Description automatically generated

FMath to control the life limits



