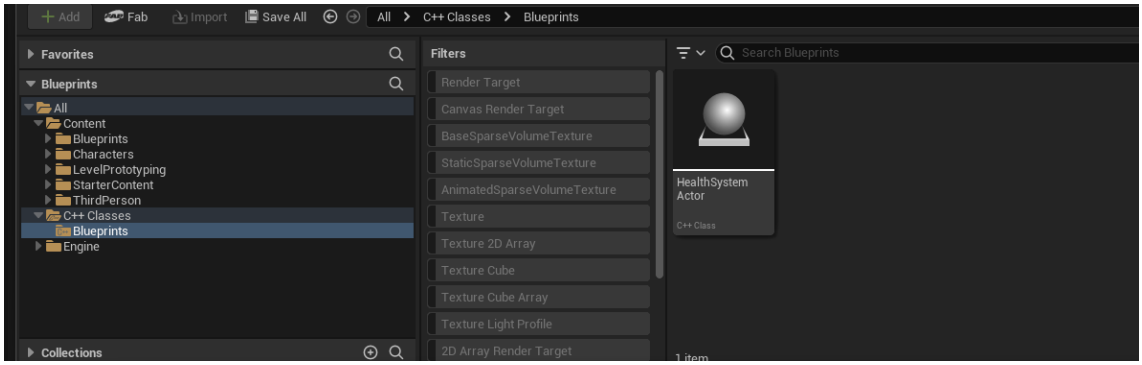
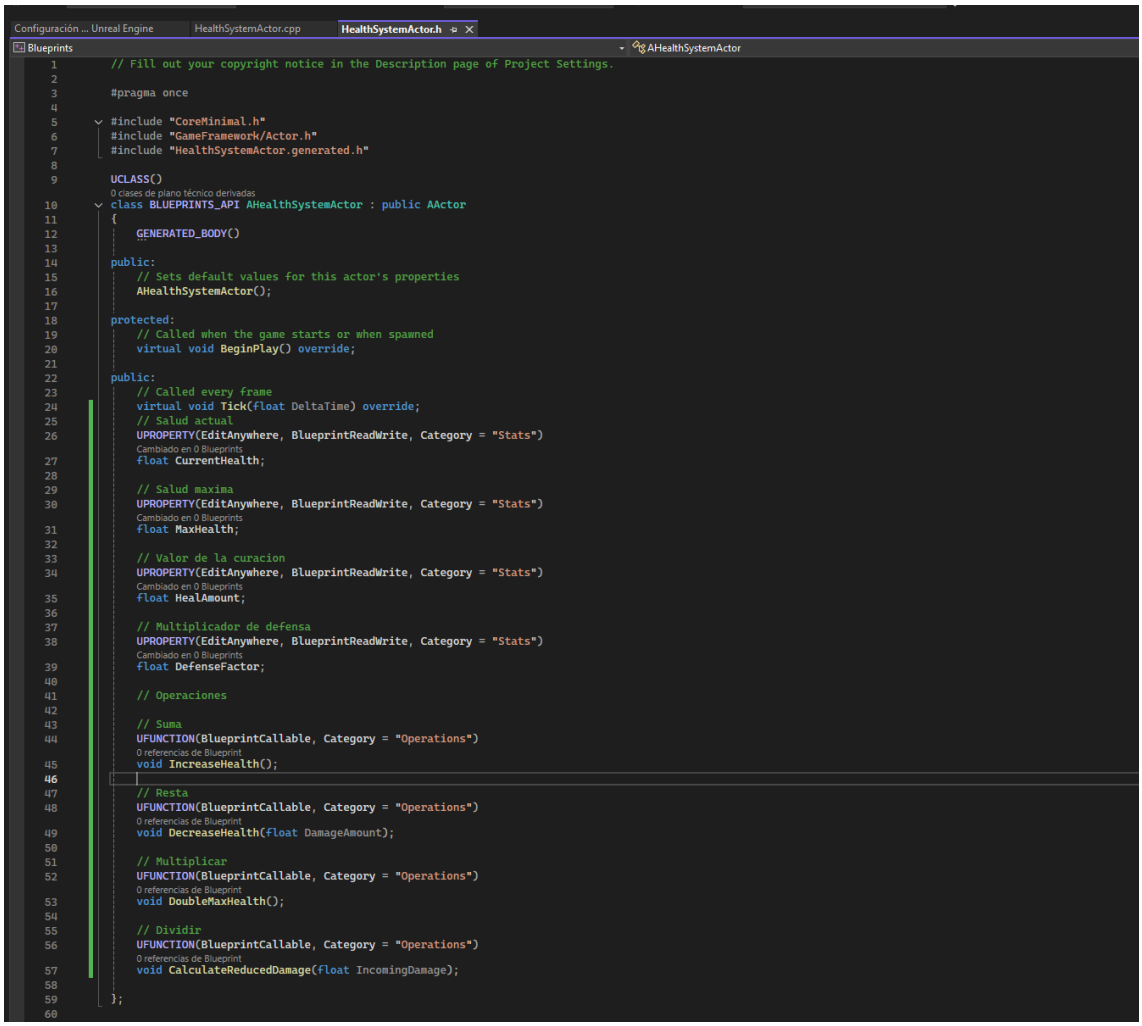
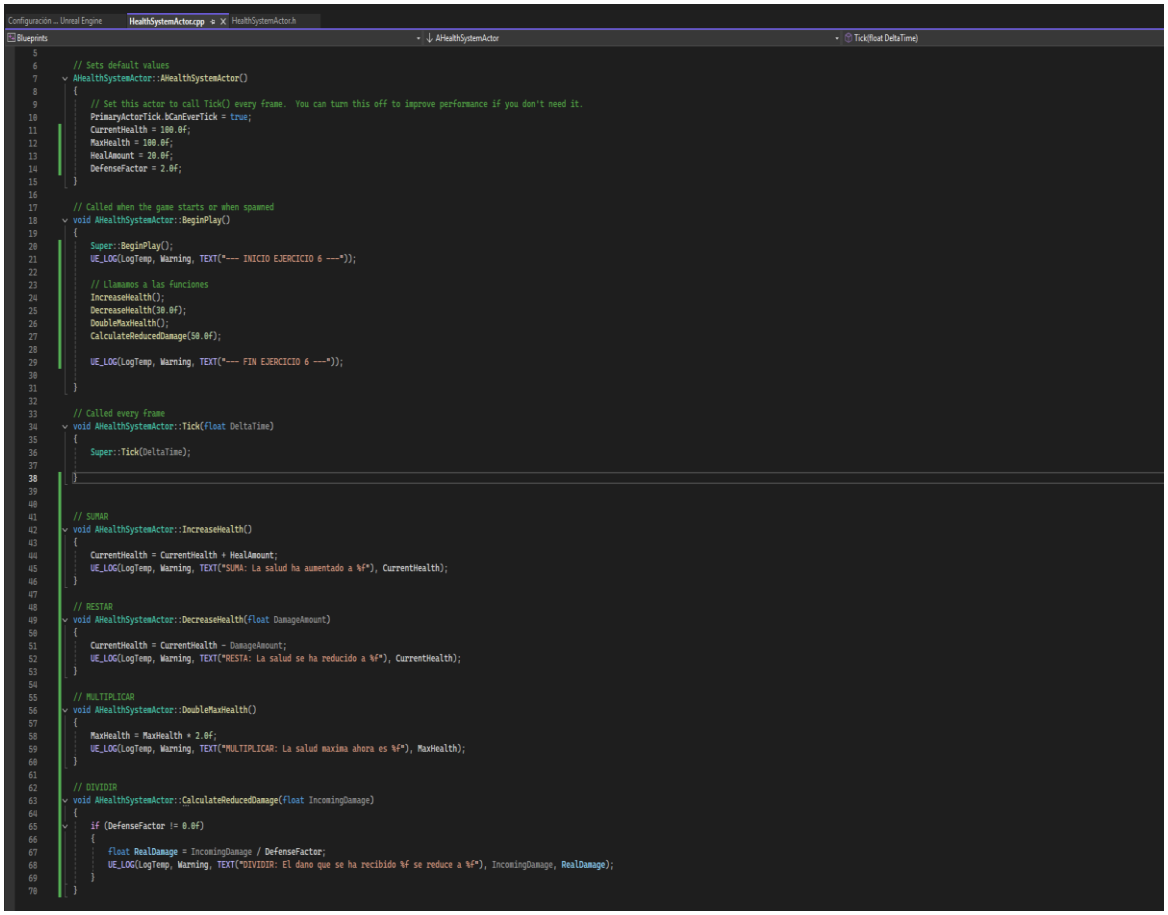
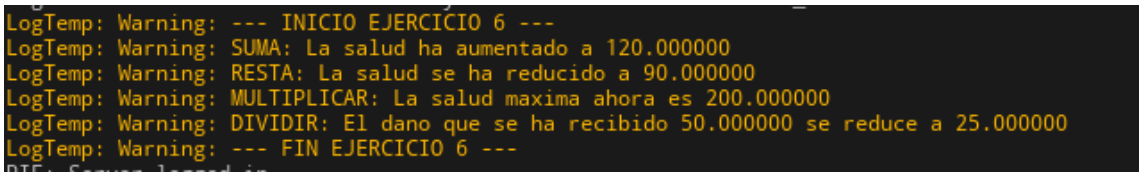

Add
Fab
Import
Save All
All
C++ Classes
Blueprints
Favorites
Blueprints
All
Content
Blueprints
Characters
LevelPrototyping
StarterContent
ThirdPerson
C++ Classes
Blueprints
Engine
Collections
Filters
Render Target
Canvas Render Target
BaseSparseVolumeTexture
StaticSparseVolumeTexture
AnimatedSparseVolumeTexture
Texture
Texture 2D Array
Texture Cube
Texture Cube Array
Texture Light Profile
2D Array Render Target
Search Blueprints
HealthSystem Actor
C++ Class
1 item


Configuración ... Unreal Engine
HealthSystemActor.cpp
HealthSystemActor.h
Blueprints
AHealthSystemActor
// Fill out your copyright notice in the Description page of Project Settings.
#pragma once
#include "CoreMinimal.h"
#include "GameFramework/Actor.h"
#include "HealthSystemActor.generated.h"
UCLASS()
0 clases de plano técnico derivadas
class BLUEPRINTS_API AHealthSystemActor : public AActor
{
GENERATED_BODY()
public:
// Sets default values for this actor's properties
AHealthSystemActor();
protected:
// Called when the game starts or when spawned
virtual void BeginPlay() override;
public:
// Called every frame
virtual void Tick(float DeltaTime) override;
// Salud actual
UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Stats")
Cambiado en 0 Blueprints
float CurrentHealth;
// Salud maxima
UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Stats")
Cambiado en 0 Blueprints
float MaxHealth;
// Valor de la curacion
UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Stats")
Cambiado en 0 Blueprints
float HealAmount;
// Multiplicador de defensa
UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Stats")
Cambiado en 0 Blueprints
float DefenseFactor;
// Operaciones
// Suma
UFUNCTION(BlueprintCallable, Category = "Operations")
0 referencias de Blueprint
void IncreaseHealth();
// Resta
UFUNCTION(BlueprintCallable, Category = "Operations")
0 referencias de Blueprint
void DecreaseHealth(float DamageAmount);
// Multiplicar
UFUNCTION(BlueprintCallable, Category = "Operations")
0 referencias de Blueprint
void DoubleMaxHealth();
// Dividir
UFUNCTION(BlueprintCallable, Category = "Operations")
0 referencias de Blueprint
void CalculateReducedDamage(float IncomingDamage);
};



```cpp
// Sets default values
AHealthSystemActor::AHealthSystemActor()
{
    // Set this actor to call Tick() every frame.  You can turn this off to improve performance if you don't need it.
    PrimaryActorTick.bCanEverTick = true;
    CurrentHealth = 100.0f;
    MaxHealth = 100.0f;
    HealAmount = 20.0f;
    DefenseFactor = 2.0f;
}

// Called when the game starts or when spawned
void AHealthSystemActor::BeginPlay()
{
    Super::BeginPlay();
    UE_LOG(LogTemp, Warning, TEXT("--- INICIO EJERCICIO 6 ---"));

    // Llamamos a las funciones
    IncreaseHealth();
    DecreaseHealth(30.0f);
    DoubleMaxHealth();
    CalculateReducedDamage(50.0f);

    UE_LOG(LogTemp, Warning, TEXT("--- FIN EJERCICIO 6 ---"));

}

// Called every frame
void AHealthSystemActor::Tick(float DeltaTime)
{
    Super::Tick(DeltaTime);

}


// SUMAR
void AHealthSystemActor::IncreaseHealth()
{
    CurrentHealth = CurrentHealth + HealAmount;
    UE_LOG(LogTemp, Warning, TEXT("SUMA: La salud ha aumentado a %f"), CurrentHealth);
}

// RESTAR
void AHealthSystemActor::DecreaseHealth(float DamageAmount)
{
    CurrentHealth = CurrentHealth - DamageAmount;
    UE_LOG(LogTemp, Warning, TEXT("RESTA: La salud se ha reducido a %f"), CurrentHealth);
}

// MULTIPLICAR
void AHealthSystemActor::DoubleMaxHealth()
{
    MaxHealth = MaxHealth * 2.0f;
    UE_LOG(LogTemp, Warning, TEXT("MULTIPLICAR: La salud maxima ahora es %f"), MaxHealth);
}

// DIVIDIR
void AHealthSystemActor::CalculateReducedDamage(float IncomingDamage)
{
    if (DefenseFactor != 0.0f)
    {
        float RealDamage = IncomingDamage / DefenseFactor;
        UE_LOG(LogTemp, Warning, TEXT("DIVIDIR: El dano que se ha recibido %f se reduce a %f"), IncomingDamage, RealDamage);
    }
}
```



```
LogTemp: Warning: --- INICIO EJERCICIO 6 ---
LogTemp: Warning: SUMA: La salud ha aumentado a 120.000000
LogTemp: Warning: RESTA: La salud se ha reducido a 90.000000
LogTemp: Warning: MULTIPLICAR: La salud maxima ahora es 200.000000
LogTemp: Warning: DIVIDIR: El dano que se ha recibido 50.000000 se reduce a 25.000000
LogTemp: Warning: --- FIN EJERCICIO 6 ---
```