

```

1 // Fill out your copyright notice in the Description page of Project Settings.
2
3 #pragma once
4
5 #include "CoreMinimal.h"
6 #include "GameFramework/Character.h"
7 #include "enemy1.generated.h"
8 UENUM(BlueprintType)
9 enum class EEnemyMovementStatus : uint8
10 {
11     EMS_Idle UMETA(DisplayName = "Idle"),
12     EMS_MoveToTarget UMETA(DisplayName = "MoveToTarget"),
13     EMS_Attacking UMETA(DisplayName = "Attacking"),
14     EMS_MAX UMETA(DisplayName = "MAX"),
15     EMS_Dead UMETA(DisplayName = "Dead")
16 };
17
18 UCLASS()
19 class PARK_API Aenemy1 : public ACharacter
20 {
21     GENERATED_BODY()
22
23 public:
24     // Sets default values for this character's properties
25     Aenemy1();
26     UPROPERTY(VisibleAnywhere, BlueprintReadOnly, Category = "Movement")
27     EEnemyMovementStatus EnemyMovementStatus;
28     FORCEINLINE void SetEnemyMovementStatus(EEnemyMovementStatus Status) { EnemyMovementStatus=Status; }
29     FORCEINLINE EEnemyMovementStatus GetEnemyMovementStatus() { return EnemyMovementStatus; }
30
31
32     UPROPERTY(VisibleAnywhere, BlueprintReadOnly, category="AI")
33     class USphereComponent* Agrosphere;
34
35     UPROPERTY(VisibleAnywhere, BlueprintReadOnly, category = "AI")
36     USphereComponent* Combatsphere;
37
38     UPROPERTY(VisibleAnywhere, BlueprintReadOnly, category = "AI")
39     class AAIController* Aicontroller;
40
41     UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Combat")
42     TSubclassOf<UDamageType> DamageTypeclass;
43     UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
44     float Health;
45     UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
46     float MaxHealth;
47     UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
48     float Damage;
49

```

```

50     bool bhasvalidtarget;
51
52     FTimerHandle AttackTimer;
53
54     UPROPERTY(EditAnywhere,BlueprintReadOnly,Category="Combat")
55     float AttackMinTime;
56     UPROPERTY(EditAnywhere, BlueprintReadOnly, Category = "Combat")
57     float AttackMaxTime;
58     FTimerHandle DeathTimer;
59     UPROPERTY(EditAnywhere,BlueprintReadWrite,Category="Combat")
60     float DeathDelay;
61
62     void disappear();
63
64     UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="Combat")
65         class UBoxComponent* CombatCollision;
66
67     UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "AI")
68         class UParticleSystem* HitParticles;
69     UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "AI")
70         class USoundCue* HitSound;
71
72     UFUNCTION(BlueprintCallable)
73     void Deathend();
74
75     void die(AActor *Causer);
76
77     virtual float TakeDamage(float DamageAmount, struct FDamageEvent      ↗
78         const& DamageEvent, class AController* EventInstigator, AActor*  ↗
79         DamageCauser) override;
80
81     bool alive();
82 protected:
83     // Called when the game starts or when spawned
84     virtual void BeginPlay() override;
85
86 public:
87     // Called every frame
88     virtual void Tick(float DeltaTime) override;
89
90     // Called to bind functionality to input
91     virtual void SetupPlayerInputComponent(class UInputComponent*      ↗
92         PlayerInputComponent) override;
93
94     UFUNCTION()
95     virtual void AgroSphereOnOverLapbegin(UPrimitiveComponent*      ↗
96         OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
97         OtherComp, int32 otherbodyindex, bool bFromSweep, const      ↗
98         FHitResult& SweepResult);
99
100    UFUNCTION()
101    virtual void AgroSphereOnOverLapend(UPrimitiveComponent*      ↗
102        OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
103        OtherComp, int32 otherbodyindex);

```

```
95
96     UFUNCTION()
97     virtual void CombatSphereOnOverLapbegin(UPrimitiveComponent*      ↗
        OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
        OtherComp, int32 otherbodyindex, bool bFromSweep, const      ↗
        FHitResult& SweepResult);
98     UFUNCTION()
99     virtual void CombatSphereOnOverLapend(UPrimitiveComponent*      ↗
        OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
        OtherComp, int32 otherbodyindex);
100
101     UFUNCTION(BlueprintCallable)
102     void MoveToTarget(class AMaincharacter* Target);
103
104     UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="AI")
105     bool bOverlappingCombatSphere;
106
107
108     UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="AI")
109     AMaincharacter* CombatTarget;
110
111     UPROPERTY(EditAnywhere,BlueprintReadWrite,Category="Combat")
112     class UAnimMontage* CombatMontage;
113
114     UFUNCTION()
115     void combatOnOverLapbegin(UPrimitiveComponent*      ↗
        OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
        OtherComp, int32 otherbodyindex, bool bFromSweep, const      ↗
        FHitResult& SweepResult);
116
117     UFUNCTION()
118     void combatOnOverLapend(UPrimitiveComponent*      ↗
        OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* ↗
        OtherComp, int32 otherbodyindex);
119
120     UFUNCTION(BlueprintCallable)
121     void activatecollision();
122
123     UFUNCTION(BlueprintCallable)
124     void deactivatecollision();
125
126
127
128     void attack();
129     UFUNCTION(BlueprintCallable)
130     void attackend();
131
132     UPROPERTY(visibleAnywhere,BlueprintReadOnly,Category="Combat")
133     bool bAttacking;
134 };
135
```