C++ : Collision으로 범위 내 객체 탐지

📌일정 범위 안에 플레이어가 들어오면 몬스터는 플레이어를 추격

1. `TArray<FOverlapResult> OverlapResults;`
    - `OverlapResults`는 충돌 결과를 저장할 배열. 이 배열은 나중에 감지된 객체들의 정보를 담는다.
2. `FCollisionQueryParams CollisionQueryParam( NAME_None , false , ControllingPawn );`
    - `CollisionQueryParam`은 충돌 쿼리 파라미터를 설정하는 객체
        - `NAME_None`은 특정 이름을 설정하지 않음을 의미
        - `false`는 단순히 이 쿼리가 복잡한 물리 시뮬레이션이나 다른 처리를 포함하지 않는다는 것을 나타낸다
        - `ControllingPawn`은 충돌 쿼리에서 무시할 객체를 지정합니다. 즉, `ControllingPawn` 자신은 충돌 검출에서 제외된다.
3. `bool bResult = GetWorld()->OverlapMultiByChannel(`
    - `GetWorld()`는 현재 월드를 가져오는 함수입니다.
    - `OverlapMultiByChannel` 함수는 주어진 충돌 채널과 모양을 사용하여 여러 객체의 충돌을 감지
        - `OverlapResults` 충돌 결과를 저장할 배열
        - `Center` 구체의 중심 위치
        - `FQuat::Identity` 회전을 의미하며, 여기서는 회전 없이 기본 상태를 사용
        - `ECollisionChannel::ECC_GameTraceChannel` 사용할 충돌 채널을 지정 (여기서는 몬스터의 채널)
        - `FCollisionShape::MakeSphere(DetectRadius)`는 반지름이 `DetectRadius`인 구체 형태의 충돌 영역을 생성

    - `CollisionQueryParam`은 앞서 설정한 충돌 쿼리 파라미터를 사용

4. `bool bResult`
    - `bResult`는 충돌 검출 결과를 나타내는 불리언 값.
    - `OverlapMultiByChannel` 함수가 하나 이상의 객체를 성공적으로 감지하면 `true`를 반환하고, 그렇지 않으면 `false`를 반환

```
void UBTService_Detect::TickNode(UBehaviorTreeComponent& Ow
nerComp, uint8* NodeMemory, float DeltaSeconds)
{
Super::TickNode(OwnerComp , NodeMemory , DeltaSeconds);
APawn* ControllingPawn = OwnerComp.GetAIOwner()->GetPawn();
if (nullptr== ControllingPawn) return;

UWorld* World = ControllingPawn->GetWorld();
if (nullptr== World) return;

FVector Center= ControllingPawn->GetActorLocation();
float DetectRadius = 600.0f;

TArray<FOverlapResult> OverlapResults;
FCollisionQueryParams CollisionQueryParam( NAME_None , fals
e , ControllingPawn );
bool bResult = GetWorld()->OverlapMultiByChannel(
    OverlapResults ,
    Center ,
    FQuat::Identity ,
    ECollisionChannel::ECC_GameTraceChannel1 ,
    FCollisionShape::MakeSphere( DetectRadius ) ,
    CollisionQueryParam
);

if(bResult)
{
    for(auto const& OverlapResult : OverlapResults)
    {
        AProjectDCharacter* player = Cast<AProjectDCharacte
```

```
r>( OverlapResult.GetActor() );
        if(player) //&&player->GetController()->IsPlayerCon
troller()
        {
            OwnerComp.GetBlackboardComponent()->SetValueAsO
bject( AMonsterAIController::TargetKey , player );

            DrawDebugSphere( World , Center , DetectRadius
, 16 , FColor::Green , false , 0.2f );
            DrawDebugPoint( World , player->GetActorLocatio
n() , 10 , FColor::Blue , false , 0.2f );
            DrawDebugLine( World , ControllingPawn->GetActo
rLocation() ,player->GetActorLocation(), FColor::Blue , fal
se , 0.2f );

        }
    }
}

DrawDebugSphere( World , Center , DetectRadius , 16 , FColo
r::Red , false , 0.2f );

}
```