**=== 네트워크 플로우 ===**

[좀비 BT 관련 네트워크 플로우]

* 이동 관련)

1. [서버] 좀비 스폰 => [클라] 스폰 좀비 객체 정보 받음
2. [서버] BT 실행 -> A\* 실행 -> (경우에따라)새로운 경로, 목표지점 => [클라] 목표지점(전체 경로를 보내지 않고 바로 다음 목표지점의 좌표), 좀비 현재 위치 정보 받고 갱신
3. 경로따라 위치 갱신 & 계산 (1초씩 ?m이동)

[서버] – A\*로 나온 경로 따라 움직임 계산 (커스텀 함수 Walk 이용)

deltaTime(경과시간) 이용하기

-> 새로운 좀비 위치 클라로 전송

[클라] – 서버로부터 받은 새로운 위치로 좀비 위치 갱신 (동기화 작용)

& 서버로부터 새로 받은 목표지점으로 향하게 움직임계산 ->

서버와 똑같은 Walk 함수 만들어서 계산하기

단, 마찬가지로 딜레이 없이 deltaTime(경과시간)에

따라 위치계산하기 (언리얼 내부 함수 Tick에서)

1. [서버] BT 다시 실행 (딜레이 시간 마다) => [클라] 새로운 경로/목표지점, 좀비 위치 정보 받고 갱신

\*\*\* {결국 2번 과정으로 돌아가서 2, 3번 과정 계속 반복}

\*\*\*\*\*{결과적으로, 좀비는 서버에서 BT가 실행되는 텀마다 각 클라들에게 좀비 정보를 갱신시켜서 결국 동기화가 되는 방식}

* BT에 Decorator에서 사용하는 bool 변수들 초기화 관련)

해당 bool 변수 목록:

CanSeePlayer, HeardShouting, HeardFootSound, KnewPlayerLocation

1. [클라] 서버의 좀비 BT에 Decorator(조건식 검사)에서 사용하는 bool 변수들에 대한 검사 독자적으로 실행

(좀비가 플레이어를 포착 중 인지 아닌지? , 샤우팅이 있었다면 샤우팅을 들었는지? , 발소리가 있었다면 발소리를 들었는지? , 플레이어를 본적이 있었는지?)

1. [서버] 클라에서 검사하여 검사 결과가 성공이 되어 해당 패킷을 보내온다면, 해당 좀비들의 패킷에 따른 bool 변수들 값 갱신
2. [서버] 해당 bool 변수값을 이용해서 BT 수행

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* {주의할 점} \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

다음과 같이 한 좀비의 상태에 대해 여러 클라에서 반복 검사하고 그 결과 값을 서버로 전송하게 되면, 서버에서 한 좀비에 대해 여러 번 그 결과값을 받고 여러 번 갱신하게 되는 문제가 발생! 따라서 검사를 하는 걸 줄이게 만들거나, (예를 들어, 플레이어를 포착하는지를 판단하는 건, 각 클라에서도 자기 자신과 근처 좀비들만 검사하고 다른 클라는 안 하도록..) 서버에서 한 좀비에 대해 여러 번의 중복된 결과값을 받는 다면, 먼저 들어온 결과값은 일단 반영하고, 일정시간을 설정하여 그 뒤에 중복되어 들어온 결과값은 무시하도록 만들어야 한다. (예를 들어, 샤우팅 좀비와 같은 경우, 각 클라이언트들의 핑 상태에 따라 샤우팅을 여러 번 서버에 부르게 될 수 있음)

* 공격 관련)

{좀비가 플레이어 공격}

1. [서버] BT에서 공격 범위 계산 검사(공격 시야각 검사까지 – 공격 반경은 짧아서 Detect 시야검사처럼 장애물검사 필요X => [X] 사실 Detect에서 이미 시야 검사를 하고 여기 Attack이 실행되므로 따로 해줄 필요는 없음!) -> 범위 안에 들면 공격 명령 전송 (전 클라이언트에게 – 애니메이션 재생 때문에)

=> [클라] 공격 명령 수신

1. [클라] 공격 수행(공격 직접 받는 클라만 피격검사 실시) -> [클라]에서 공격 수행 후 피격 검사 수행, 결과 서버로 전송 => [서버] 피격 검사 결과 수신 -> 검사 결과 반영 -> (클라이언트가 죽었을 때 전 클라이언트에게) => [클라] 죽은 플레이어 정보 수신