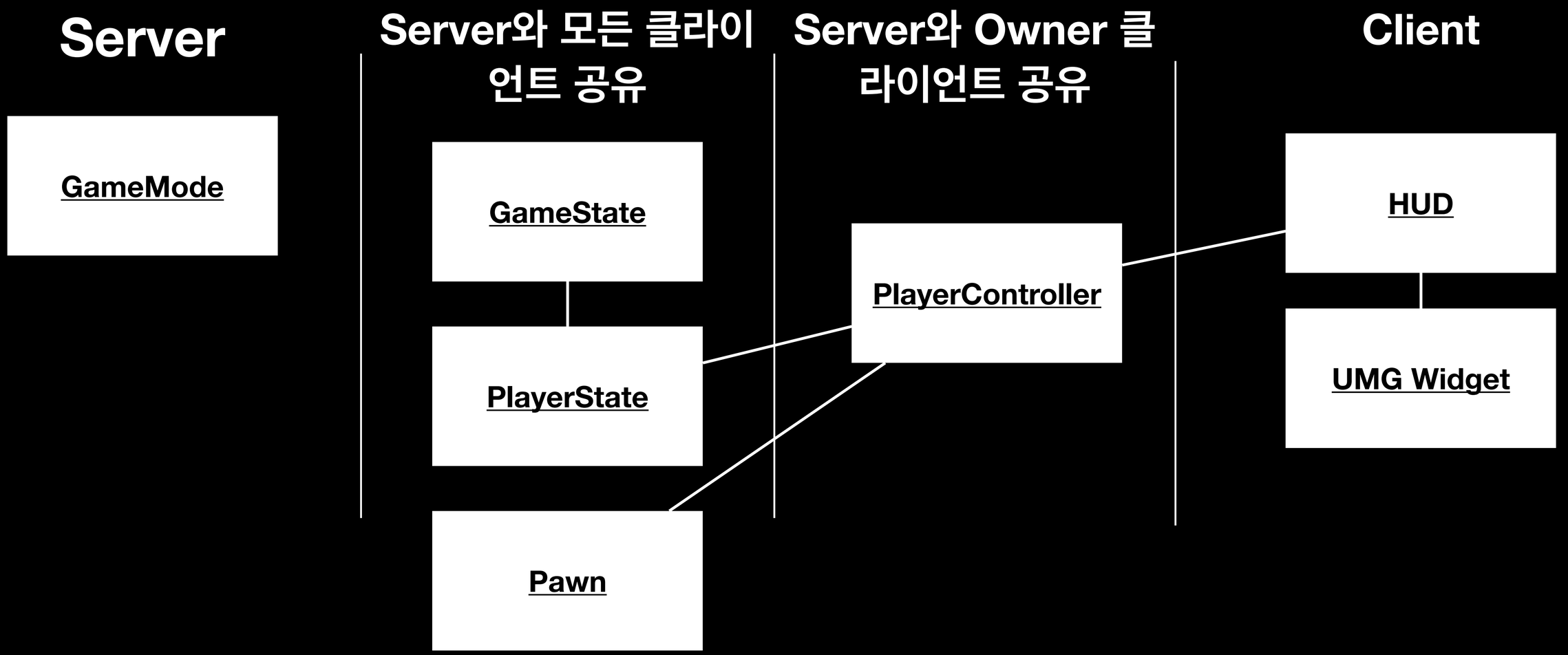
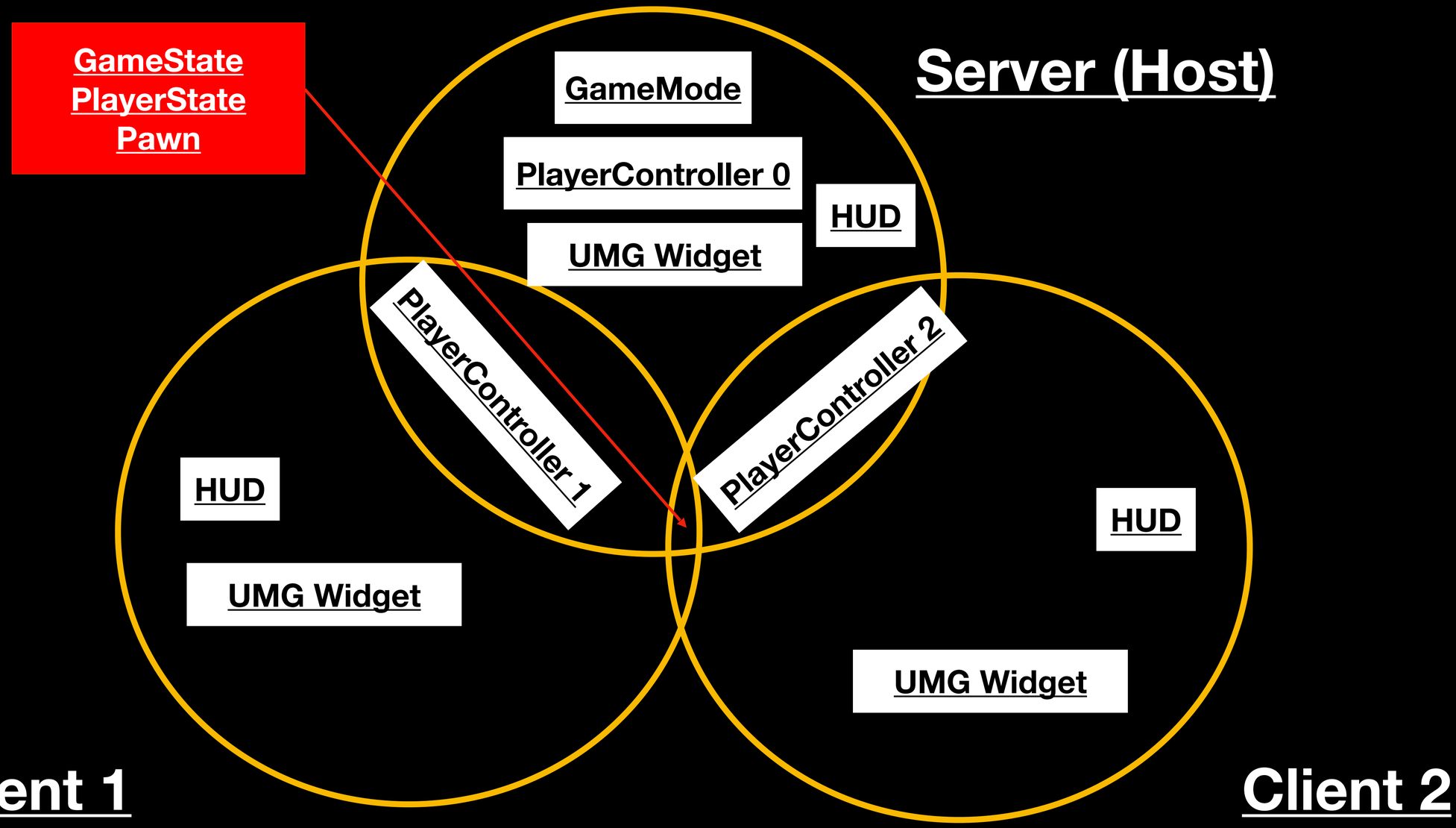
네트워크 Crash Course

github.com/Unreal-Engine-Developers-Korea

네트워크 프레임워크


Server
Server와 모든 클라이언트 공유
Server와 Owner 클라이언트 공유
Client
GameMode
GameState
PlayerState
Pawn
PlayerController
HUD
UMG Widget

Listen Server 네트워크 프레임워크



OnRep + RPC 호출 조건

1. Actor 또는 ActorComponent에서만 호출 가능 (즉 UWidget, AnimInstance, 등에서 호출 불가능)
2. 해당 Actor/ActorComponent Replicate 활성화 되어야함
3. RPC를 호출하는 Actor의 Owner는 Instigator 플레이어야함
    a. 누가 Owner이냐가 중요하다는 소리임

Ownership (소유권)

Server에서 RPC 호출할 때

액터 소유권	Replicate 안됨	Multicast	Server	Client
미소유	Server에서 실행	서버 + 모든 클라이언트	서버에서 실행	서버에서 실행
Client 소유	Server에서 실행	서버 + 모든 클라이언트	서버에서 실행	액터의 소유 클라이언트
Server 소유	Server에서 실행	서버 + 모든 클라이언트	서버에서 실행	서버에서 실행

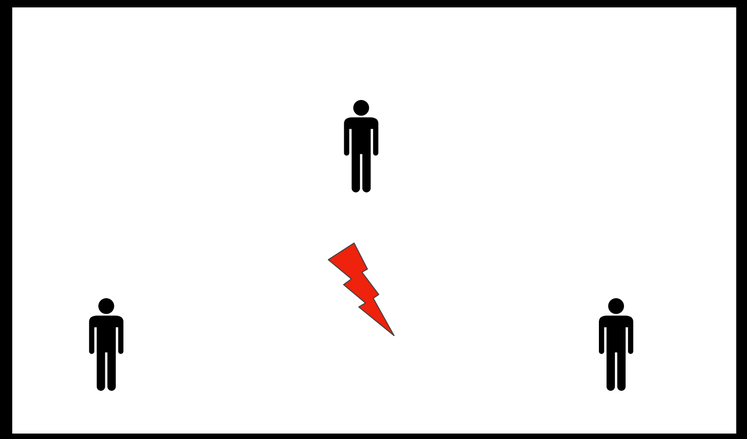
Ownership (소유권)

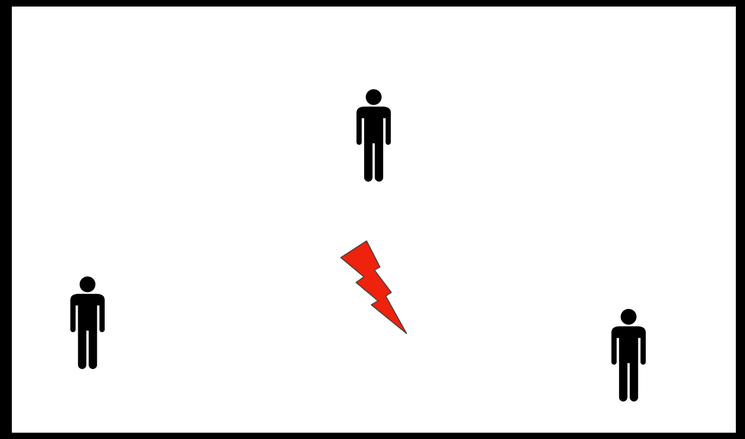
Client에서 RPC 호출할 때

액터 소유권	Replicate 안됨	Multicast	Server	Client
미소유	호출하는 클라이언트	호출하는 클라이언트	드랍	호출하는 클라이언트
호출하는 클라이언트	호출하는 클라이언트	호출하는 클라이언트	서버에서 실행	호출하는 클라이언트
다른 클라이언트	호출하는 클라이언트	호출하는 클라이언트	드랍	호출하는 클라이언트
Server 소유	호출하는 클라이언트	호출하는 클라이언트	드랍	호출하는 클라이언트

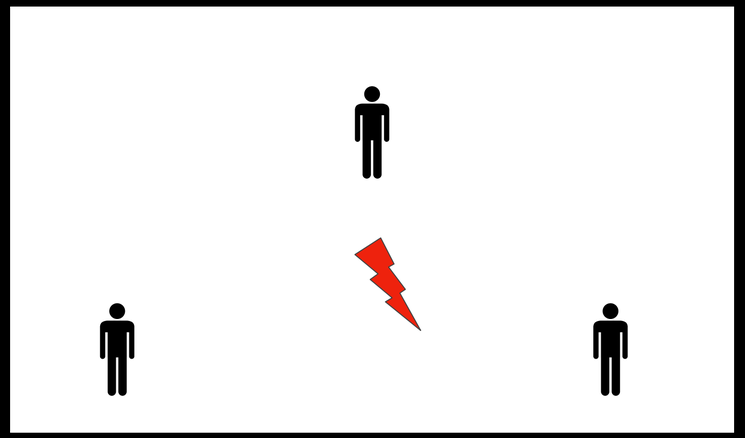
Ownership (소유권)

Server (Host)





Client 1



Client 2

서버에서만 가능한 작업

1.Spawn
2.Destroy
3.Attach
    a.(예외 Owner이면 가능)

Client가 Spawn, Destroy 권한이 없는 이유

자주 햇갈리는 부분

- **Server World + Server Player**
  - Listen Server에서는 Server == Server 플레이어 (Host)
  - Dedicated Server에서는 서버 플레이가 존재하지 않음
- **Multicast || OnRep을 하든 둘다 Server에서 업데이트를 해줘야하기 때문에 Server RPC를 해줘야함**

OnRep 주의할 점

- OnRep은 Client에서 자동으로 호출되지만 Server에서 자동으로 호출이 안 됨 -> 강제로 해야함
- Replicated 변수를 바꿀때 Server에서 바꿔야함 -> HasAuthority가 아니면 Server RPC로 해주면 된다
- OnRep은 변수가 바뀔때만 호출이됨

OnRep vs RPC

OnRep/ Replicated 변수

- Stateful (관리형 또는 저장식) Event
  - 이전 이벤트 또는 사용자 상호 작용의 정보를 유지한다는 의미
- 예) 플레이어 순서, 체력, 탄약

RPC

- Transient (과도 過渡 또는 소멸식) Event
  - 한 **순간에** 새로운 상태로 옮아가거나 바뀌어 간다는 의미
- 예) VFX, SFX, UI

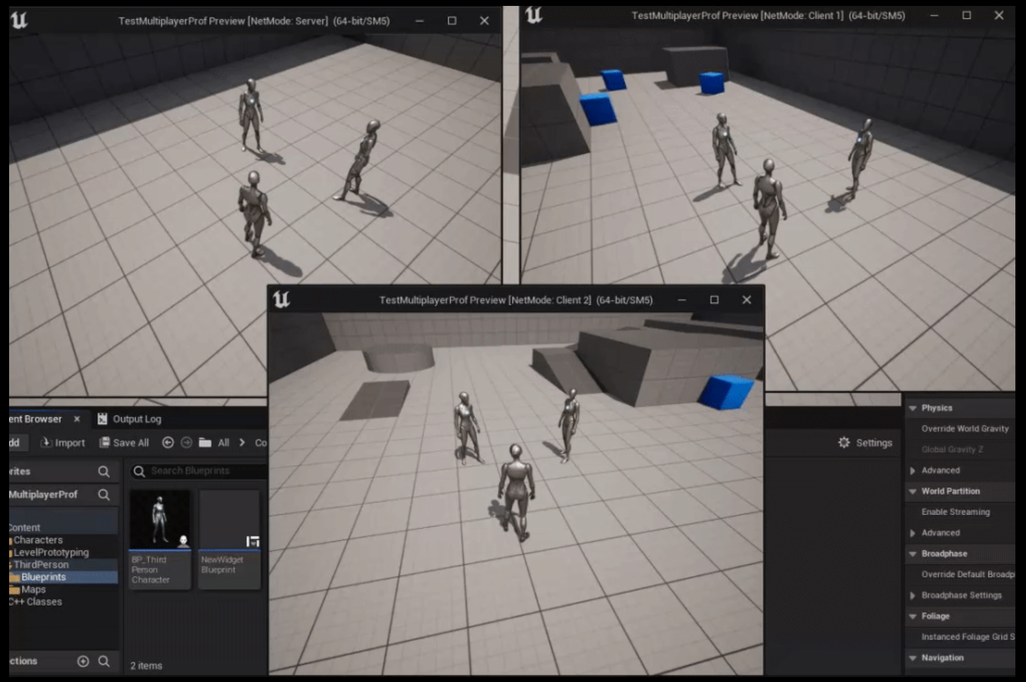
OnRep vs RPC

한 질문으로 정리할 수 있음:

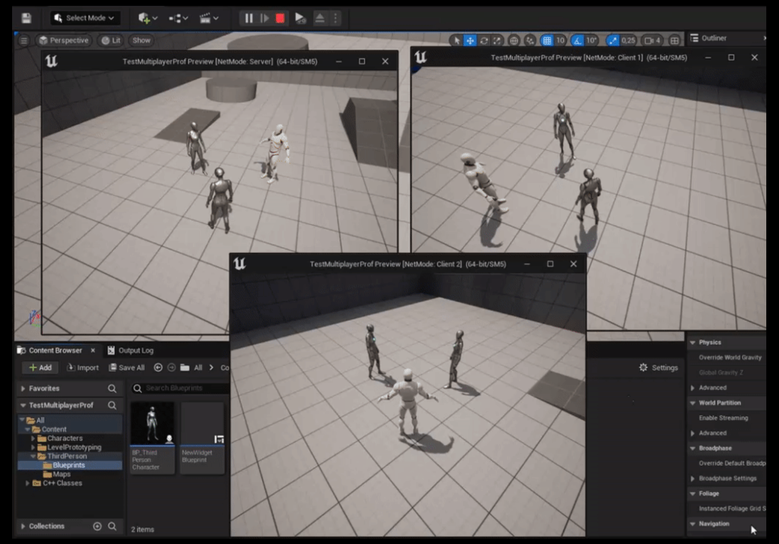
새로운 플레이어가 세션을 접속하면 동기화를 해야하는 기능인가?

True: OnRep **False: RPC**

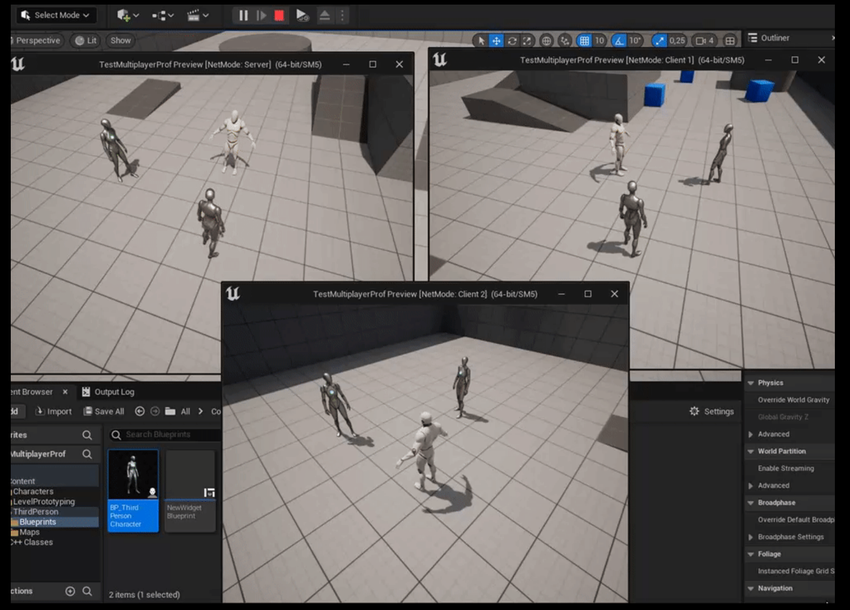
OnRep 예시



Change Mesh



Multicast RPC



OnRep

퀴즈

애니메이션은 OnRep일까 아니면 RPC일까??

애니메이션

상황에 따라 다름ㅎ

애니메이션
RPC의 경우

- Transient Event가 호출될 때
  - 애니메이션 몽타쥬 (Attack, etc.)
    - 점프
    - Emote
    - Etc.

애니메이션

Stateful Event는?

애니메이션

Replication의 경우

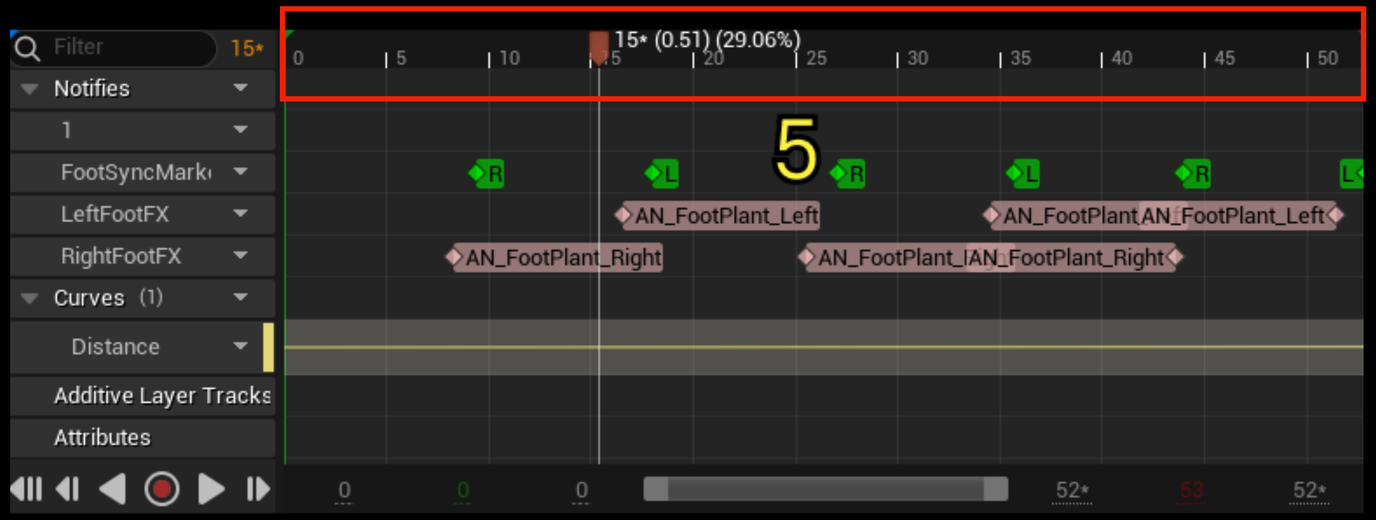
- Stateful Event가 호출될 때 (State Machine과 관련된 것들)
  - Crouch
  - Gun/Rifle Animation

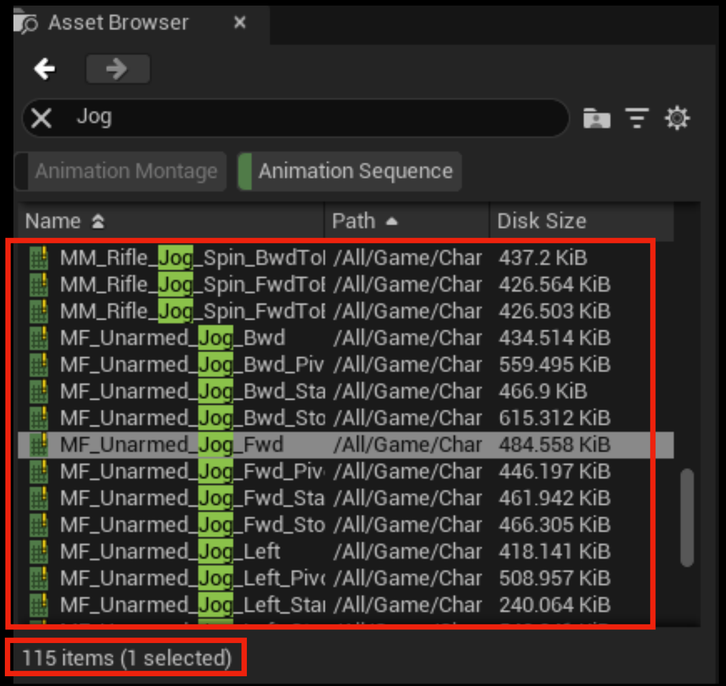
- 근데 이건 함정질문이었음ㅎ
  - 애니메이션 자체를 Replicate은 안함

애니메이션

그냥 AnimInstance을 Replicate 하면 안되나요?

- 모든 State Machine 안에 있는 애니메이션을 Replicate하려고 하면 그자리에서 해고;;
  - 1 애니메이션 = N Bone 포즈 x 프레임 수 x 그외 데이터
  - 115 애니메이션 Replicate = 넥슬라이스





애니메이션

그래서... Skeletal Mesh 포즈랑 AnimInstance는 Replicate을 안함

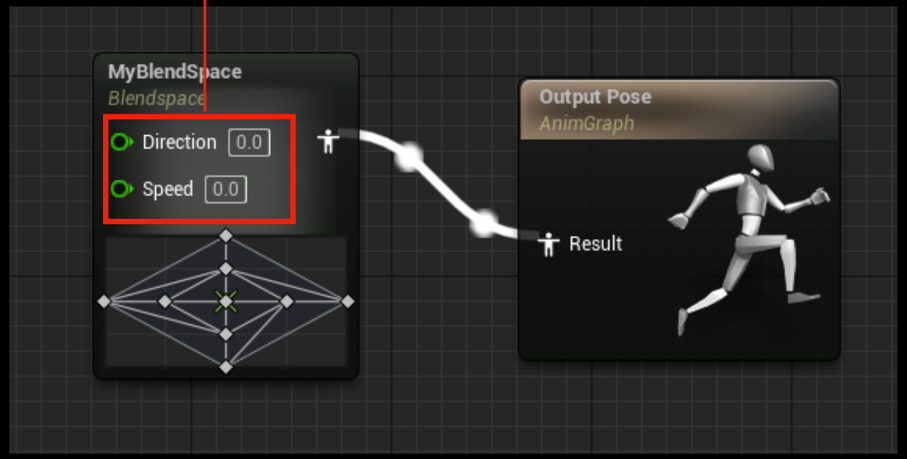
애니메이션

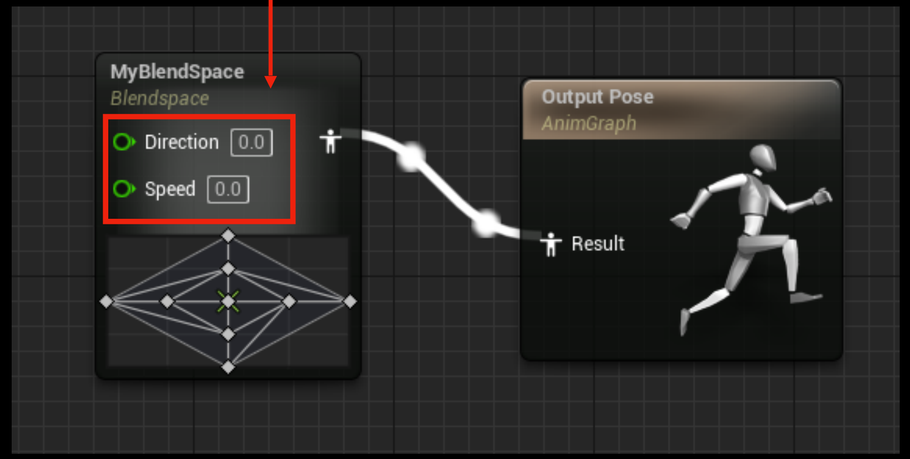
동기화 해결책

- Pawn의 애니메이션 상태머신을 제어할 속성(변수)만 Replicate해주면 됨

Replicate

나머지는 Client Character가 알맞은 애니메이션 적용





Server World의 Client 1

나머지 World의 Client 1

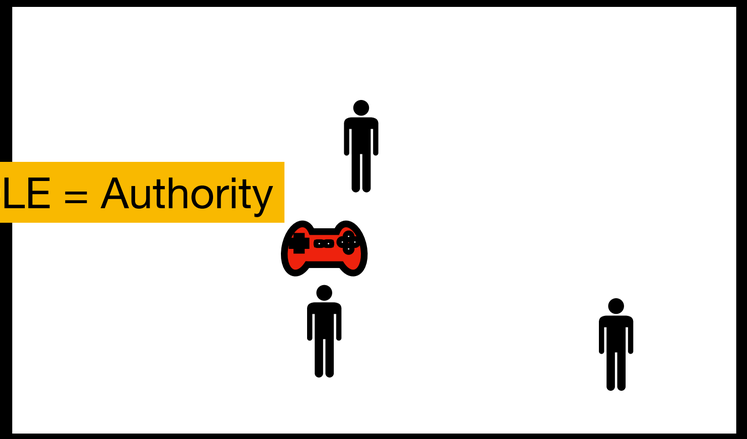
애니메이션

- 따라서 애니메이션 상태머신을 제어할 변수들만 있으면 됨
- Float Direction, Speed, Acceleration, etc…
- Bool bDead, bCrouch, bRunning, etc…

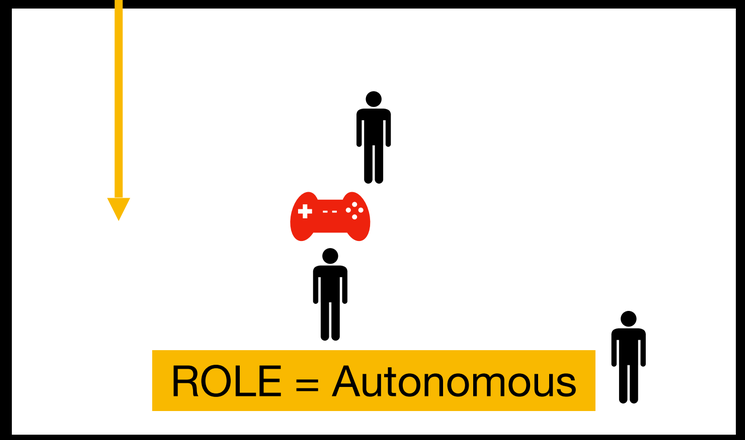
애니메이션 동기화 과정 (Listen Server)

I. Replicated 변수 할당
Direction = …
Acceleration = …
Speed = …

Server (Host)

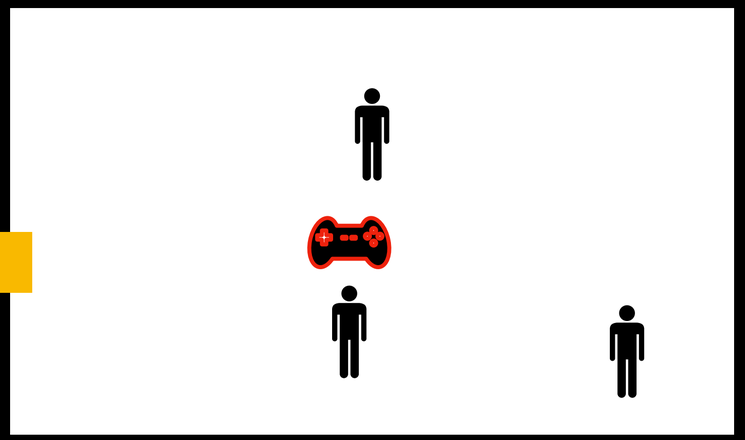


1. 로컬 컨트롤러 Input



Client 1

ROLE = Simulated



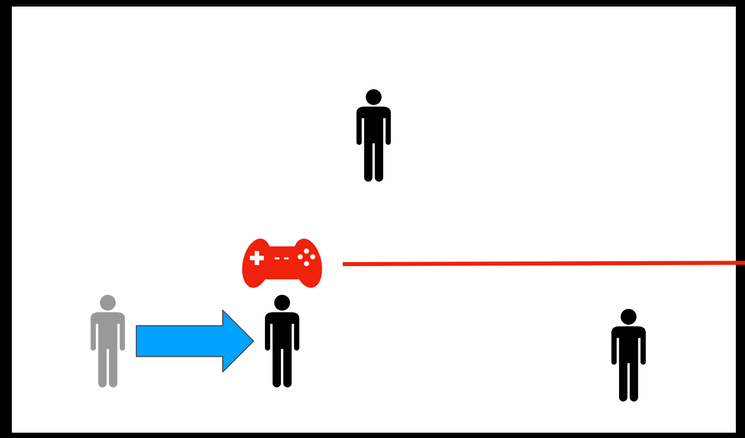
Client 2

애니메이션 동기화 과정 (Listen Server)

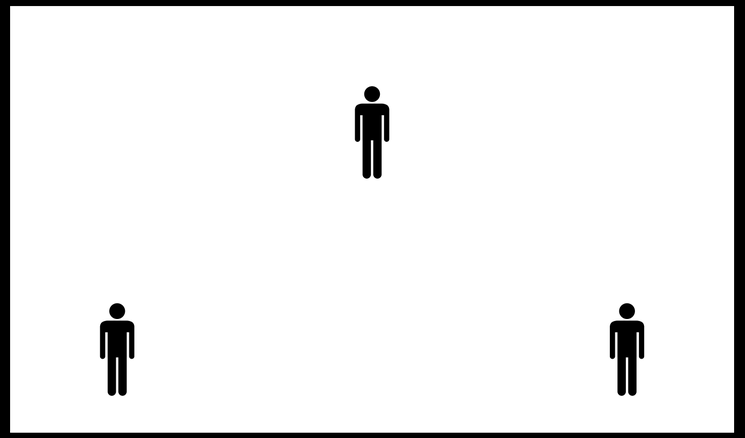
Server (Host)

II. Replicate한 변수를 AnimInstance에 적용하여 애니메이션 실행

2-A. Client 1 캐릭터 이동



2-B. Client 1 Authority 컨트롤러로 전달



Client 1

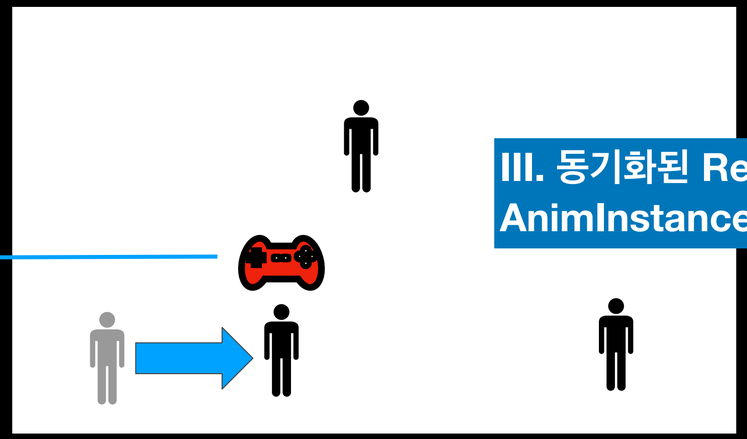
Client 2

애니메이션 동기화 과정 (Listen Server)

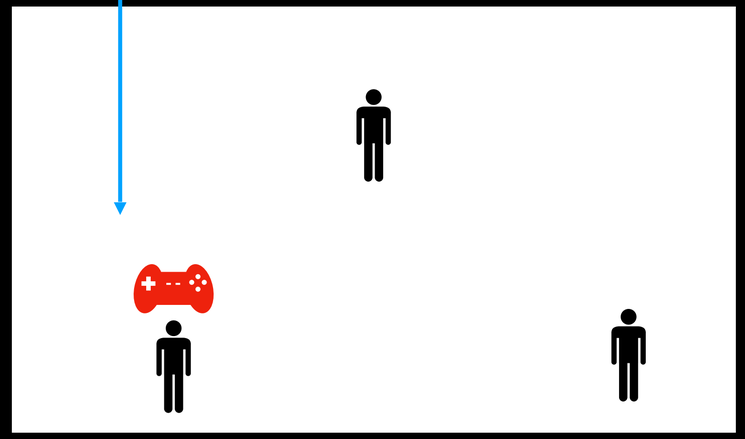
3-A. Server World에서 Client 1 Character 이동

3-B. Server World의 Client 1 Character 랑 Client 1 World의 Character 위치가 일치한지 확인

Server (Host)

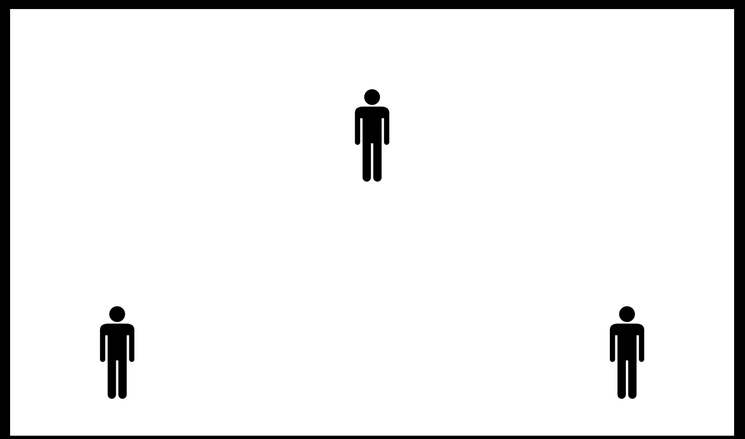


III. 동기화된 Replicated된 변수를 AnimInstance에 적용 -> 애니메이션 실행



3-C. 위치가 일치하면 Client 1 이동 허용

3-C. 위치가 일치않으면 Server World의 Client 1 Character 위치로 덮어쓰기

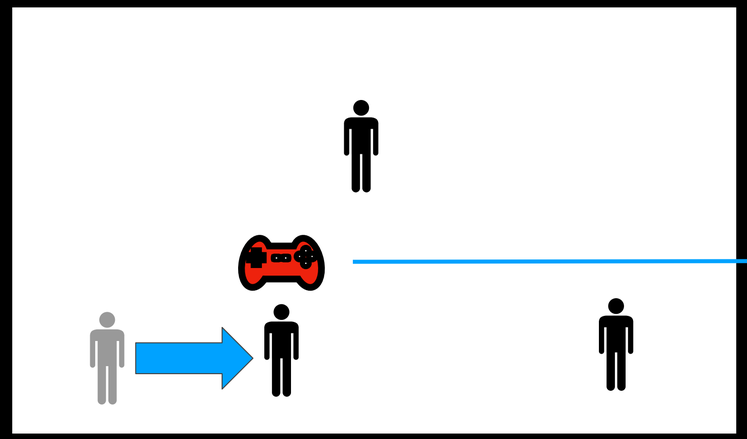


Client 1

Client 2

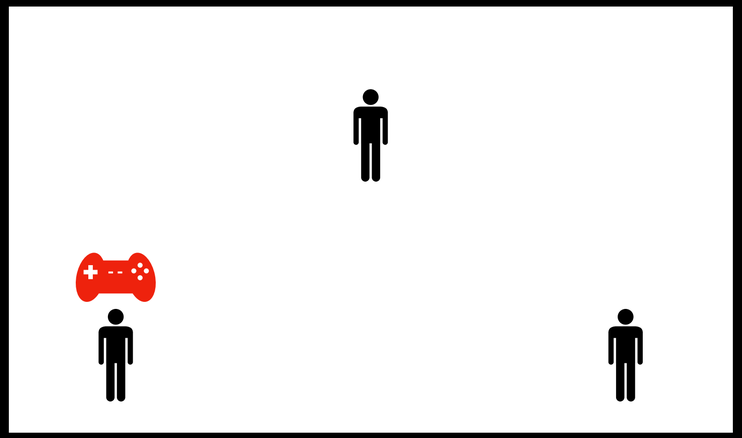
애니메이션 동기화 과정 (Listen Server)

Server (Host)

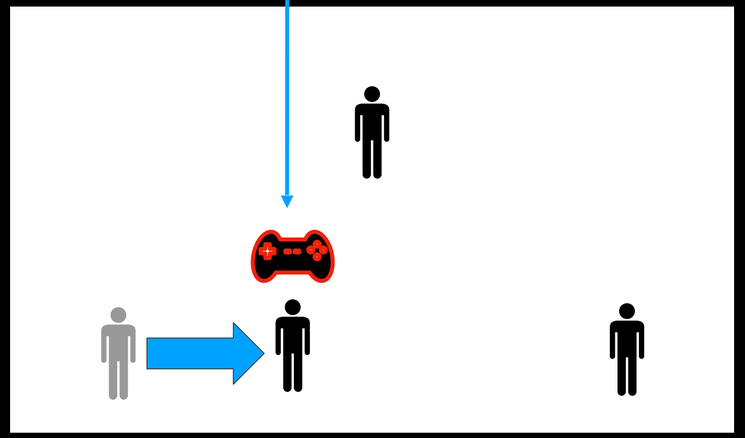


4. 나머지 Simulated Controller에게 정보 전달

IV. 동기화된 Replicated된 변수를 AnimInstance에 -> 애니메이션 실행



Client 1



Client 2

동기화 예시 #1

Mario Kart 팀: Overlap

Overlap 로직 플로우

1.BeginOverlap 호출

2.Overlap한 플레이어의 UI에 변화 적용 (Star UI)

3.Actor Destroy

Overlap 로직 플로우

1. **BeginOverlap 호출 (플레이어의 모든 Instance - 모든 World에 있는 플레이어의 Character 사본 + 원본)**
   a. Instigator = 모든 World에 있는 Overlap 플레이어
   b. Effector = 모든 World에 있는 Actor
2. **BeginOverlap 호출한 플레이어에게 UI 적용 (로컬 플레이어)**
   a. Instigator = 로컬 Actor
   b. Effector = 로컬 플레이어
3. **Actor Destroy**
   a. Instigator = 서버 Actor
   b. Effector = 모든 World에 있는 해당 Actor

다른 World의 나와 로컬 나를 필터링

ServerRPC? 굳이 안해도됨

Overlap 로직 플로우

```cpp
void AMyActor::OnBeginOverlap(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor,
    UPrimitiveComponent* OtherComp, int32 OtherBodyIndex, bool bFromSweep, const FHitResult& SweepResult)
{
    APawn* OtherPawn = Cast<APawn>(OtherActor);
    if(OtherPawn){
        //로컬 플레이어
        if(OtherPawn->IsLocallyControlled()){
            //OtherPawn의 UI 변화적용

            //Destroy
        }
    }
}
```

Overlap 로직 플로우

```
void AMyActor::OnBeginOverlap(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor,
    UPrimitiveComponent* OtherComp, int32 OtherBodyIndex, bool bFromSweep, const FHitResult& SweepResult)
{
    APawn* OtherPawn = Cast<APawn>(OtherActor);
    if(OtherPawn){
        //로컬 플레이어
        if(OtherPawn->IsLocallyControlled()){
            //OtherPawn의 UI 변화적용


        }
        //Destroy
    }
}
```

동기화 예시 #2

Pokemon 팀: Spawn Pokemon

Spawn Pokemon 로직 플로우

1.플레이어가 UI로 상호작용
2.MonsterBall Spawn하기
3.VFX 적용
4.해당 포켓몬 Spawn하기
5.플레이어 + 포켓몬 애니메이션 적용
6.MonsterBall Destroy

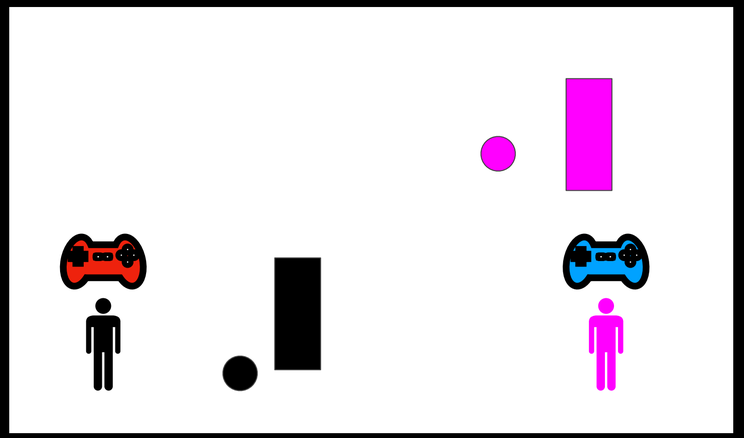
Spawn Pokemon 로직 플로우

1. **플레이어가 UI로 상호작용 (로컬 플레이어)**
2. **MonsterBall Spawn하기 (Instigator: 로컬 플레이어, Effector: 모든 월드)**
    a. 서버 RPC. Why? Spawn은 서버에서만 가능
3. **VFX 적용 (Instigator: 로컬 플레이어, Effector: 모든 월드)**
    a. Multicast RPC. Why? 모든 월드가 Effector임
4. **해당 포켓몬 Spawn하기**
    a. 서버 RPC
5. **플레이어 + 포켓몬 애니메이션 적용 (Instigator:)**
    a. Multicast RPC. Why? 몽타쥬
6. **MonsterBall Destroy**
    a. ServerRPC. Why? Destroy는 서버에서만 가능

Pokemon Team 예시

Server (Host)

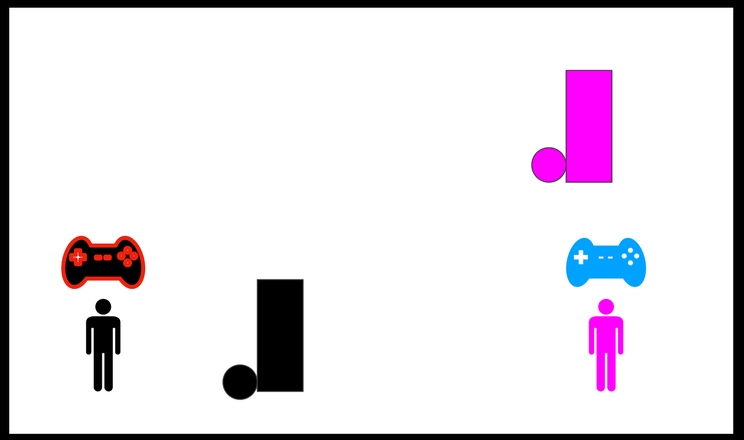
ROLE = Authority



ROLE = Authority

Client

ROLE = Simulated



ROLE = Autonomous