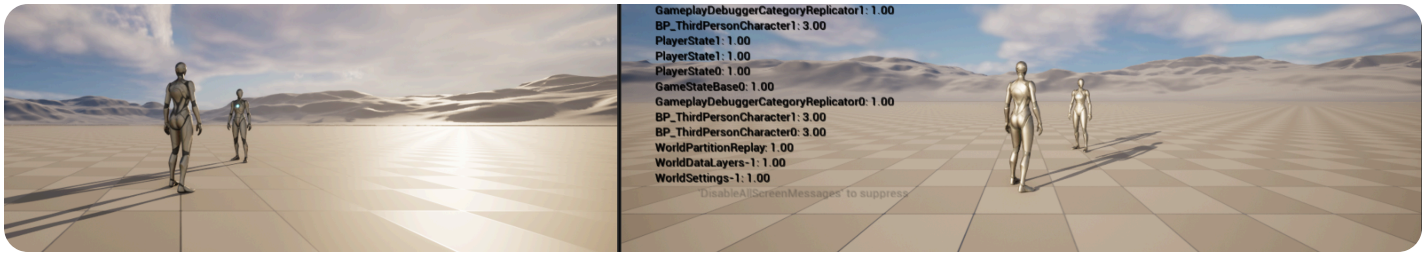
Actor Priority

Determine the network priority of an actor for replication.



Unreal Engine does not guarantee that all actors replicate during a network update. This is due to the restricted nature of networking resources. A primary limiting factor is a connection's bandwidth. The *bandwidth* of a connection is the maximum data transfer capacity of that connection. A connection becomes *saturated* when the connection exceeds its capacity. When a connection is saturated, Unreal Engine's replication system uses a load-balancing technique that assigns all actors a numerical **priority**. This priority gives each actor a fair share of the available network bandwidth resources based on how important the actor is to gameplay. Actors with higher relative priority are those that are more important to replicate, so they receive more bandwidth for replication.

Obtain an Actor's Priority

Each actor has a floating-point `AActor::NetPriority` property. Higher `NetPriority` corresponds to more bandwidth for the current actor relative to others. For example, an actor with `NetPriority == 2.0` is given more resources than an actor with `NetPriority == 1.0`. The only thing that matters with priorities is their ratio; you cannot improve Unreal Engine's network performance by scaling all actors' net priorities.

As a baseline, here are the initial values used by some common Unreal Engine classes:

Class	Priority
`AActor`	`1.0`
`APawn`	`3.0`
`APlayerController`	`3.0`

`NetPriority` is a baseline used for low bandwidth or saturated connections. `AActor::GetNetPriority` determines an actor's current priority based on a number of factors, including base `NetPriority`, distance to viewer, and time since last replicated.

Retrieve Actor's Current Priority

The network driver determines an actor's current priority for replication to a particular connection with a call to `GetNetPriority`. This is handled automatically by the network driver.

Override Actor Relevancy

You can customize actor priority by overriding the virtual function `GetNetPriority` in your `AActor`-derived class as well as changing the base network priority with `NetPriority`.

> Use caution when overriding an actor's `GetNetPriority`. This may have unintended consequences if you are not familiar with Unreal Engine's replication system.

How Priority is Determined

An actor's current network priority is calculated based on the time since the actor was last replicated, as well as a variety of additional factors, to obtain a floating-point priority.

Parameters

Actor network priority is based on the following input parameters:

Parameter	Description
`ViewPos`	Position of the viewer.
`ViewDir`	Direction the viewer is facing.
`Viewer`	Network object owned by the client for whom network priority is being determined. This is usually a Player Controller.
`ViewTarget`	Actor currently viewed or controlled by `Viewer`. This is usually a Pawn.
`InChannel`	Channel on which this actor is being replicated.
`Time`	Time since this actor was last replicated.
`bLowBandwidth`	True if the viewer has low bandwidth.

Priority Logic

Most of the work of `AActor::GetNetPriority` is done to compute a multiplicative factor for the constant `AActor::NetPriority` based on distance from and sightlines to the Viewer as well as the time since the current actor last replicated.

Network priority is determined as follows:

- If both of the following conditions hold, then the current actor uses its owner's network priority.
  - The current actor has an owner.
  - The current actor is set to use its owner's network relevancy.
- If at least one of the following conditions holds, then the current actor's network priority increases.
  - The current actor is the current connection's pawn.

    - The current connection's pawn is the instigator of some action.
- If neither of the previous two points are the case, then distance based calculations are performed to determine the current actor's network priority:
    - If the current actor is in front of the viewer, then priority decreases inversely proportional to set distances.
        - If the distance between the current actor and the viewer is greater than `CLOSEPROXIMITY` but less than `NEARSIGHTTHRESHOLD`, then the priority is multiplied by `0.2`.
        - If the distance between the current actor and the view is greater than `NEARSIGHTTHRESHOLD`, then the priority is multiplied by `0.4`.
    - If the distance between the current actor and the viewer is less than `FARSIGHTTHRESHOLD` and the viewer is looking at the current actor, then the priority is multiplied by `2.0`.
    - If the distance between the current actor and the view is greater than `MEDSIGHTTHRESHOLD`, then the priority is multiplied by `0.4`.

The distance and sight threshold constants have the following values:

Constant	Value
`CLOSEPROXIMITY`	`500`
`NEARSIGHTTHRESHOLD`	`2000`
`MEDSIGHTTHRESHOLD`	`3162`
`FARSIGHTTHRESHOLD`	`8000`

These constant definitions can be found in `NetworkingDistanceConstants.h`.

Priority Reference

Functions

Name	Description
`GetNetPriority`	Used to prioritize actors when deciding which actors to replicate.
`GetReplayPriority`	Similar to `GetNetPriority`. Used for prioritizing actors while recording a replay.

Properties

Name	Description
`bNetUseOwnerRelevancy`	If this actor has a valid owner, call the owner's `IsNetRelevantFor` and `GetNetPriority`.
`NetPriority`	Priority for this actor when checking for replication in a low bandwidth or saturated situation. Higher priority means it is more likely to replicate.