Documentation

By Nerulith

Changelog

Version 1.3 – May 2025

Multiplayer Support
Dedicated & Listen Server Support: Full multiplayer support now available with both dedicated and listen server modes.

Replication Modes:
Independent Replication: Each client streams and loads levels independently.
⚠️*Note: The host sees all players' generated levels. Other players only see their own.*

Shared Replication: All clients share the same level generation, visible to everyone in real time.

Player Visibility
Hide Out-of-Range Players: Players located in unloaded level chunks will be hidden from others, improving performance and preventing visual inconsistencies in per-client streaming.

Level Cleanup
Automatic Chunk Cleanup: Unused level chunks are automatically removed after a set time to reduce memory usage and improve performance.

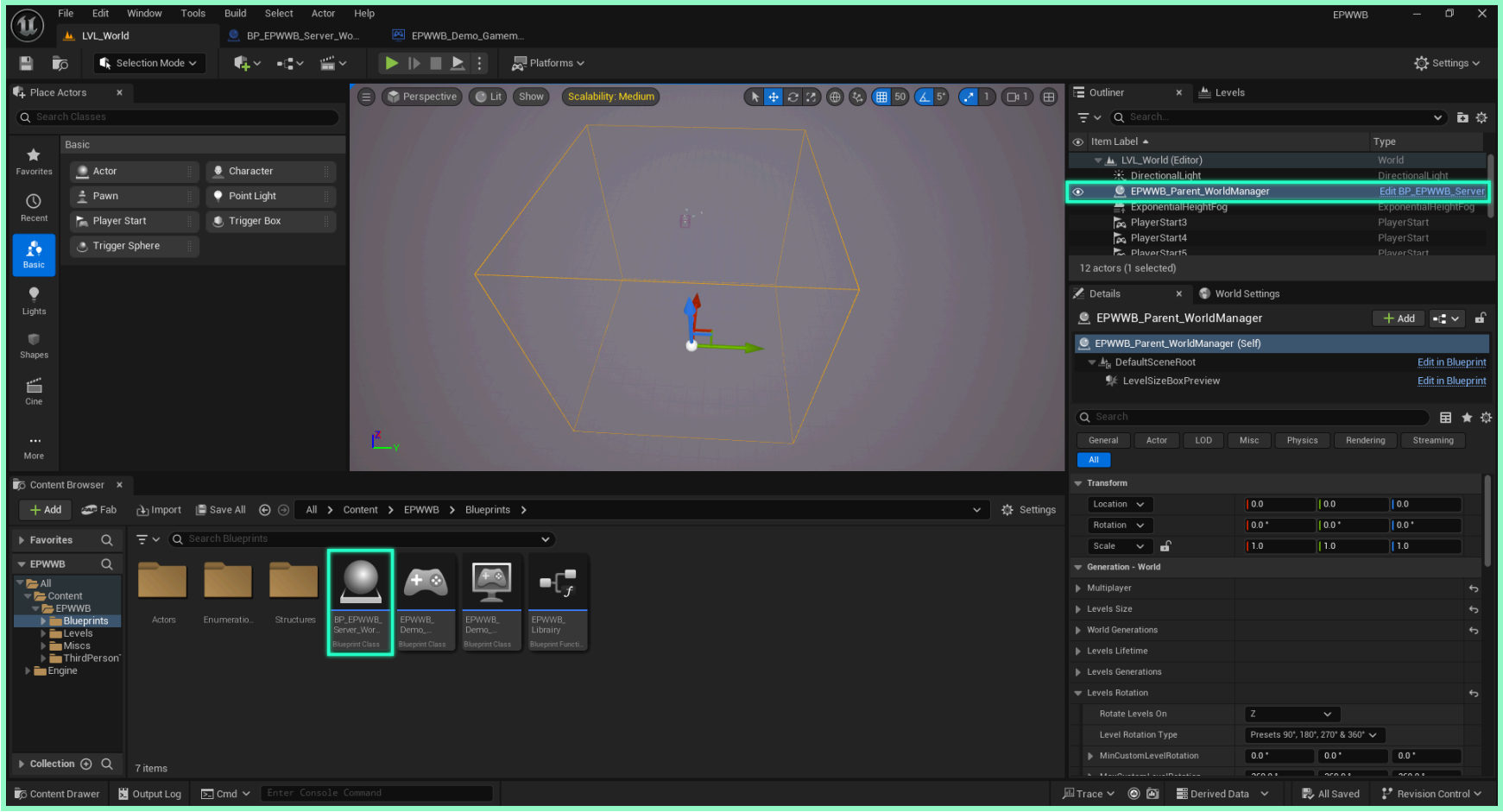
Bug Fixes
3D Perlin Noise Fix: Corrected a bug in the 3D Perlin noise.

Code Optimization
General performance and readability improvements.

Contents

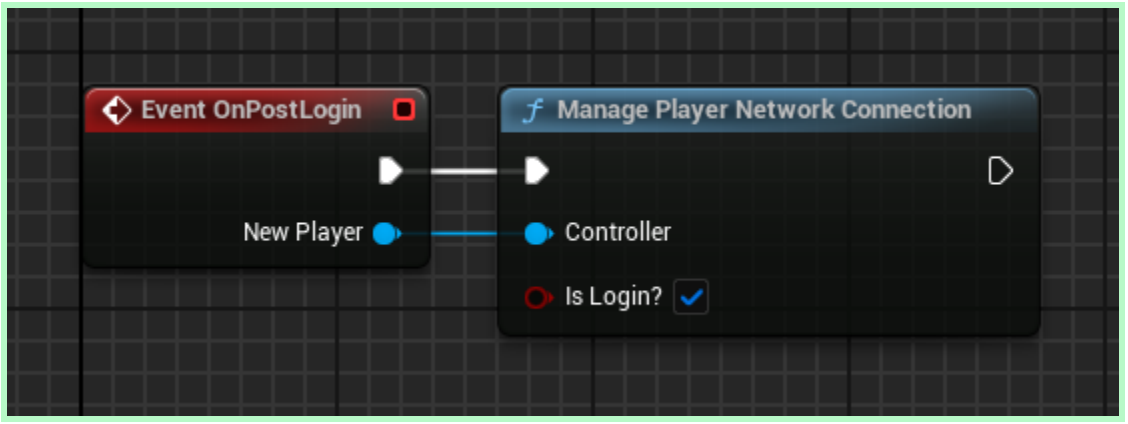
Setup

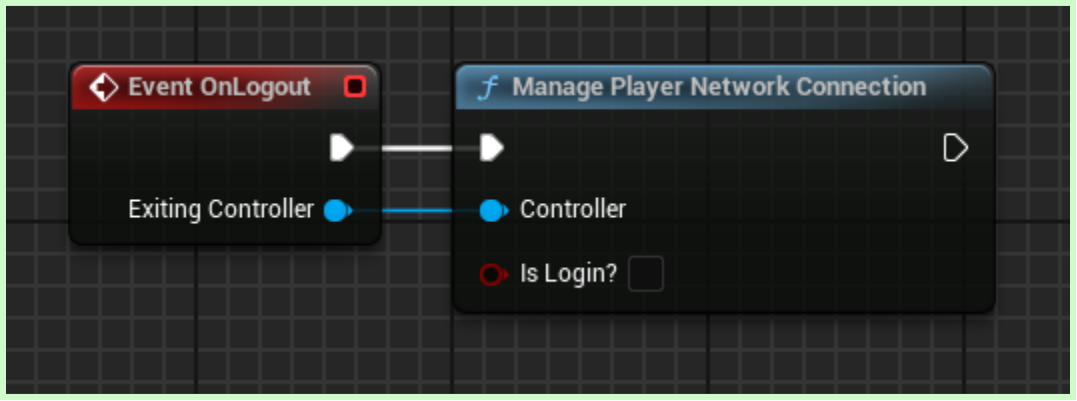
To start, simply drag and drop the **BP_EPWWB_Server_WorldManager** actor into your scene.



Next, you'll need to manage player connections. For this, you can either create your own **GameMode** or use the provided Example GameMode **EPWWB_Demo_Gamemode**.

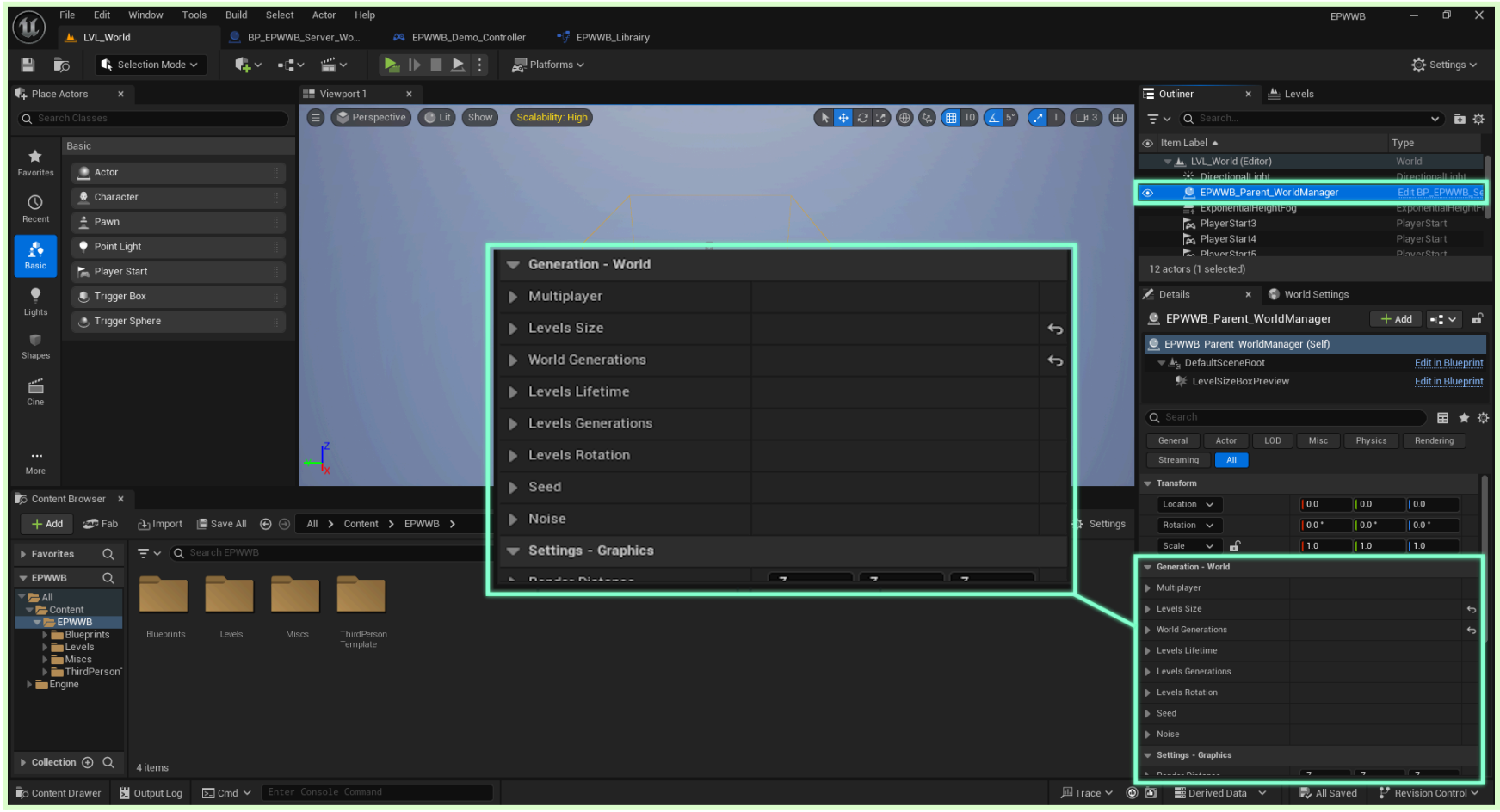
Then, in the Event Graph of your GameMode, call the **Event OnPostLogin** node.
From there, call the **Manage Player Network Connection** function.
Connect this function to the execution pin of OnPostLogin, and make sure to pass the **New Player** reference to the **Controller** input of the function, and set **IsLogin** to true.



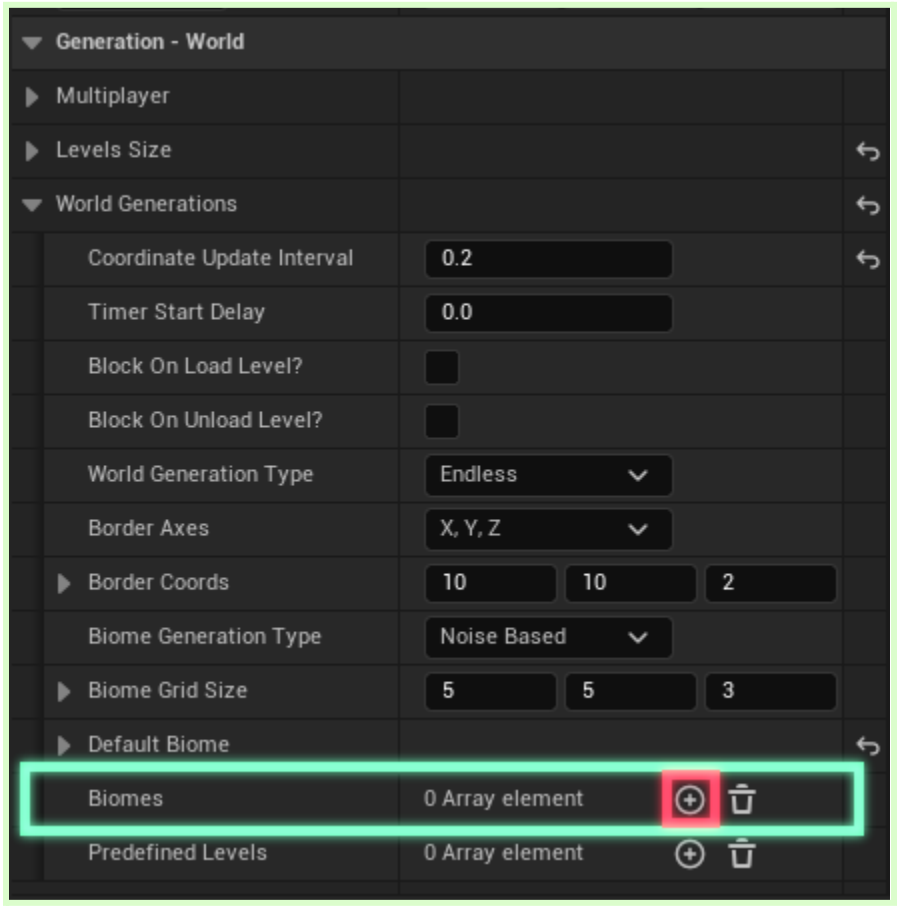


You should also handle player disconnection using the **Event OnLogout** node in the GameMode. Just like with OnPostLogin, call the **Manage Player Network Connection** function, connect it to the execution pin of **OnLogout**, and pass the **Exiting Controller** to the **Controller** input of the function, and set **IsLogin** to false.

After setting up the **GameMode**, simply click on the **BP_EPWWB_Server_WorldManager** you previously added into your scene.
In the **Details** panel, you'll find all the necessary settings to configure world generation.



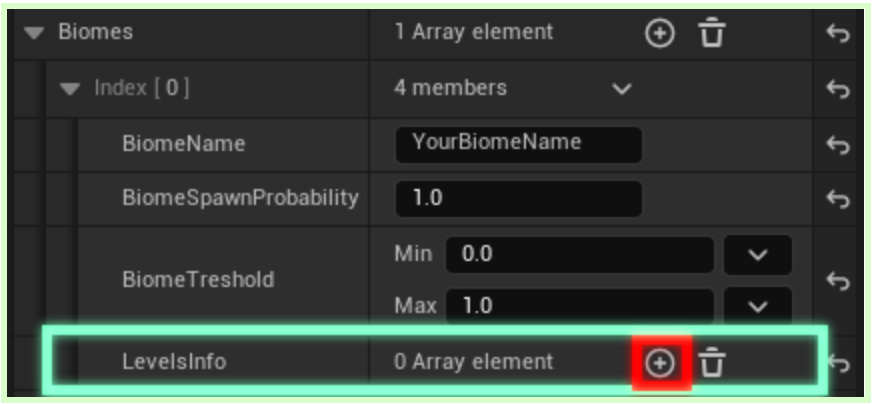
Then, navigate to the **World Generation** section and look for the **Biomes** property. Click the **plus (+)** button to add a new biome to the array.

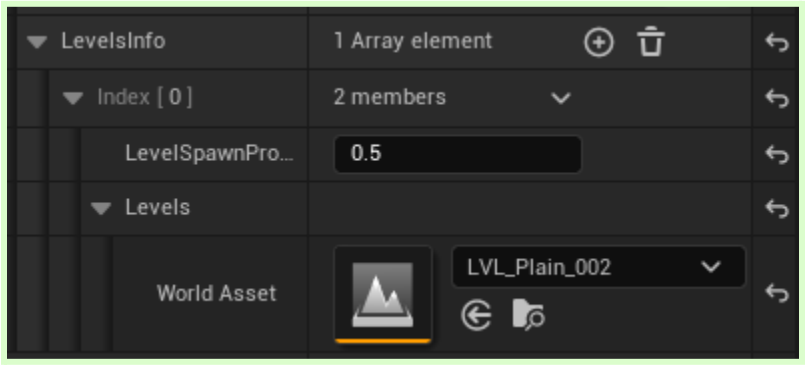


Once you've added a new biome, you can define its **name**, its **spawn probability**, and its **threshold**, which determines at what noise value the biome should appear.

Then you need to add **levels** to the biome.
Each biome contains its own list of levels, and you can either use **existing levels** or create your **own custom ones**.

To do this, look for the **LevelsInfo** property and click the **plus (+)** button to add a new level to the array.

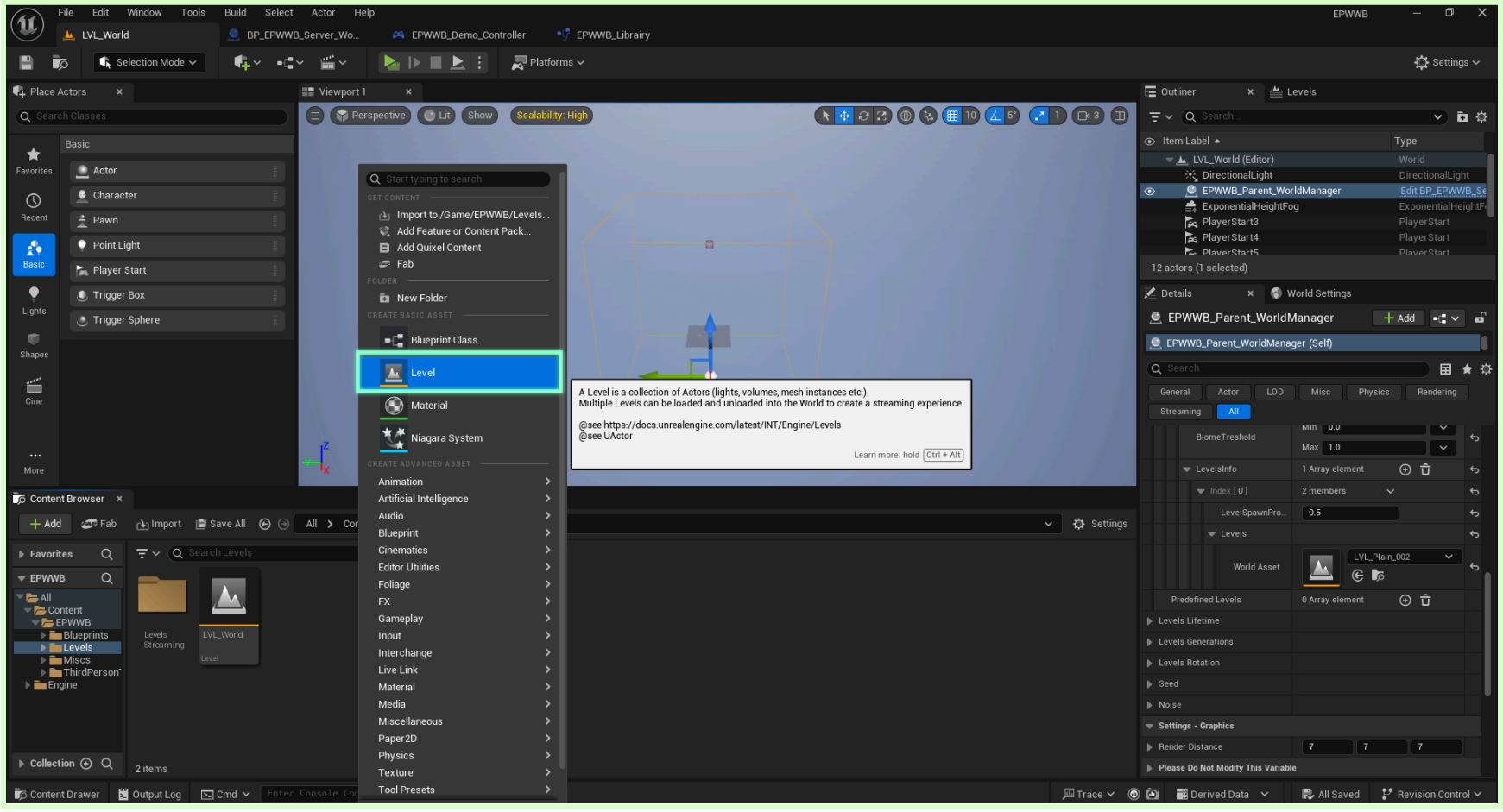




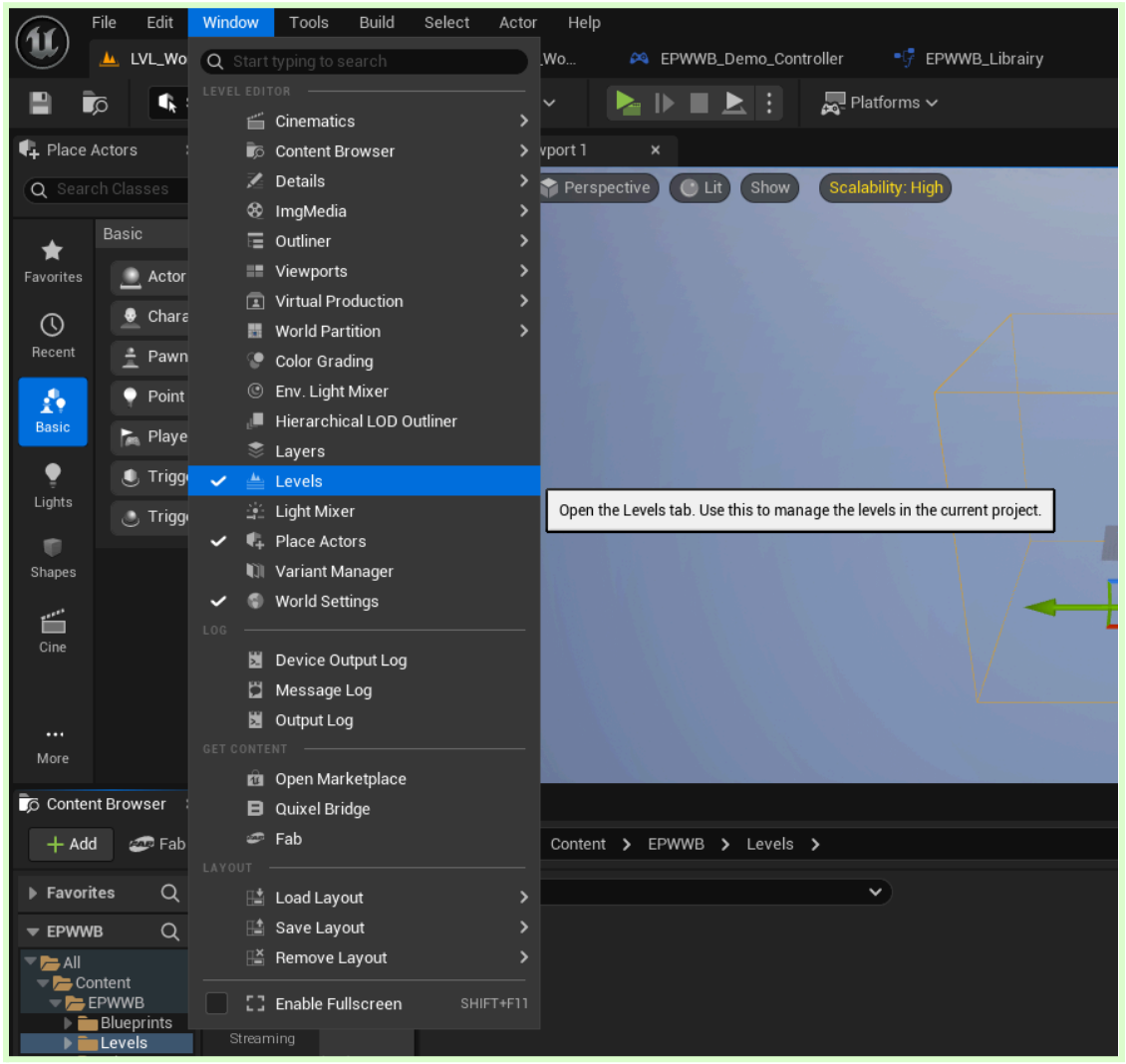
Then you can define the **spawn probability** of the level within the biome, and set the **reference to your level asset**.

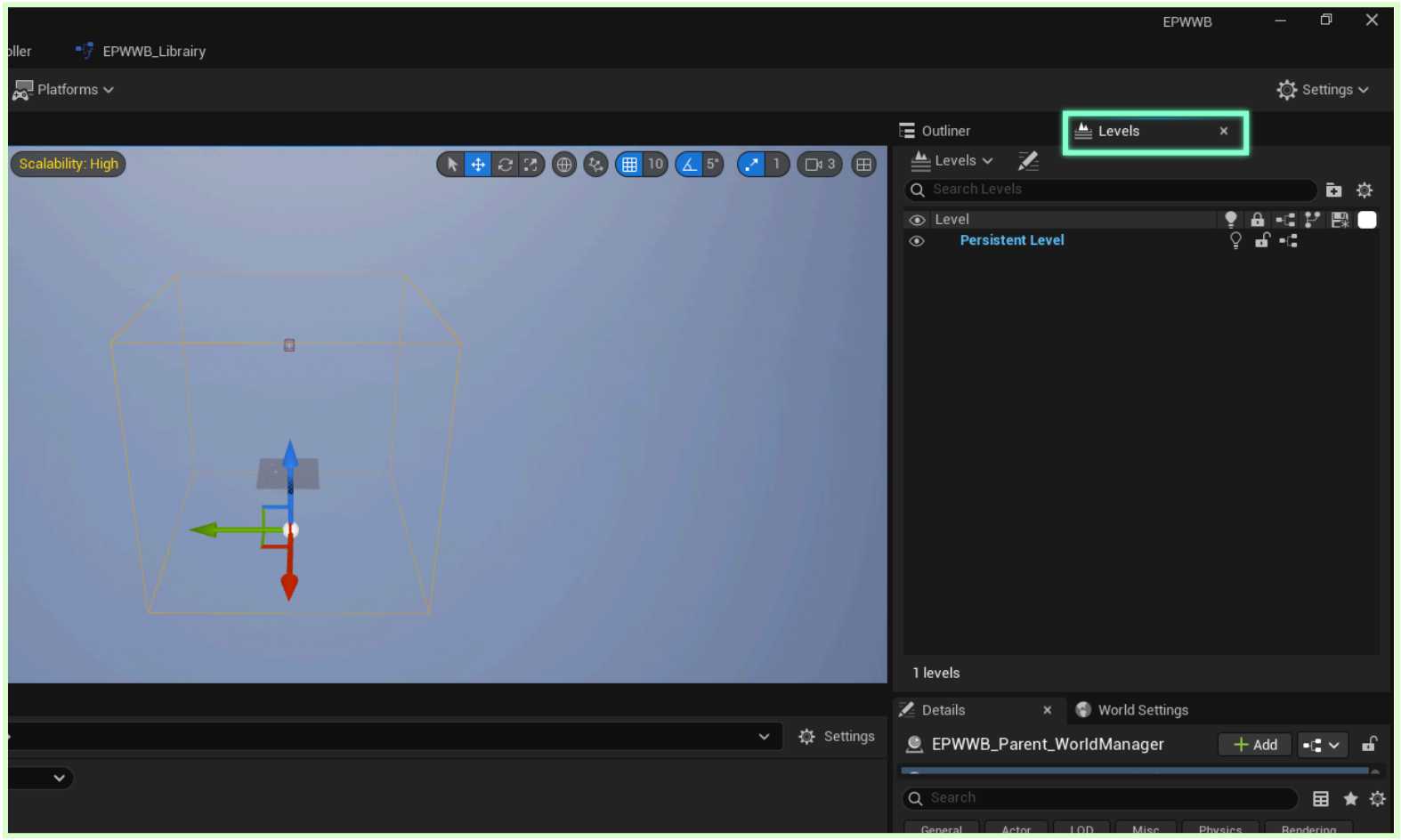
Create Your Own Levels

To create your own levels, **right-click** in the **Content Browser**, then select **"Level"** to create a new level asset.

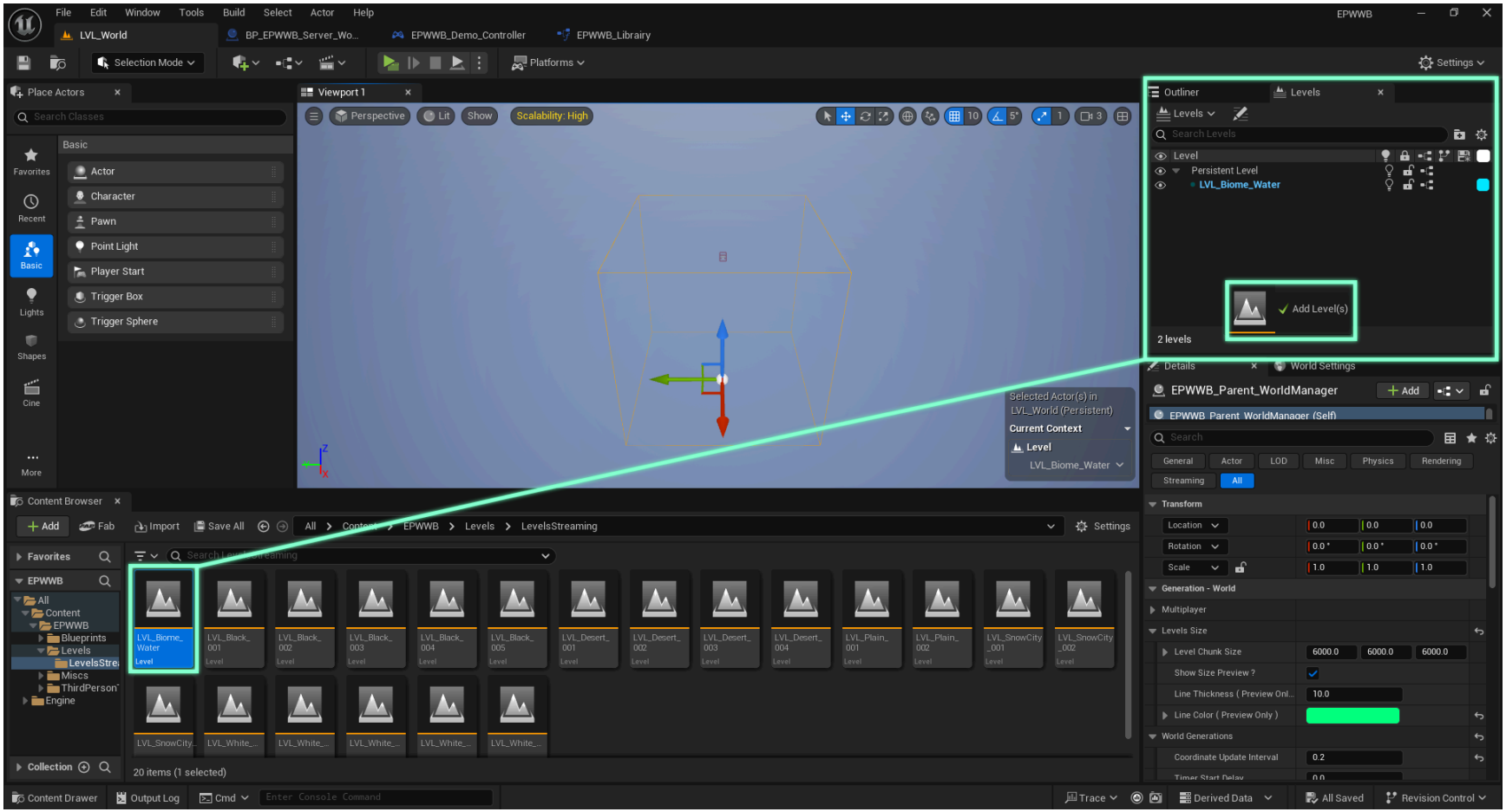


Once you've created the level, click on the **Levels** tab. If it's not visible, you can enable it by clicking on **Window** in the **top-left corner**, then selecting **Levels**.



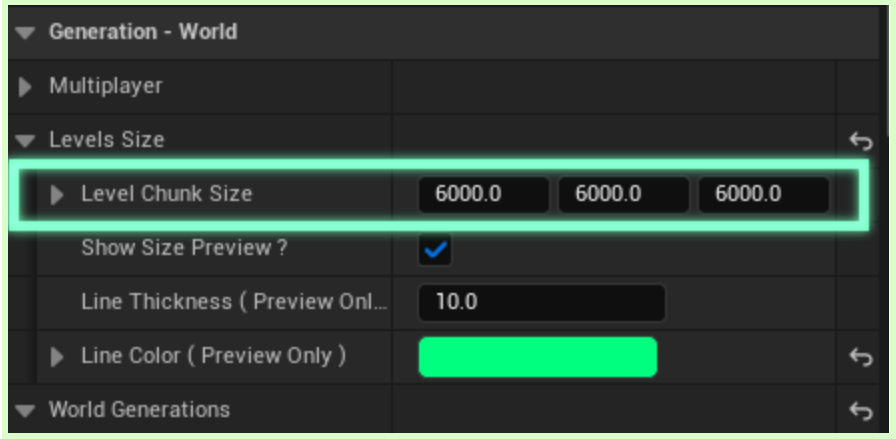

EPWWB
EPWWB_Librairy
Platforms
Settings
Scalability: High
Outliner
Levels
Search Levels
Level
Persistent Level
1 levels
Details
World Settings
EPWWB_Parent_WorldManager
Add
Search
Settings

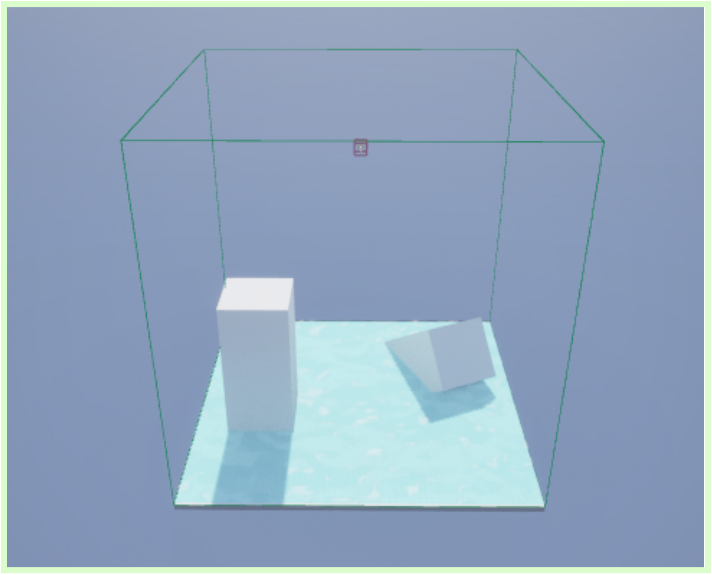
Next, drag the **level** you just created into the **Levels** tab. Make sure to select it to ensure that any modifications you make are applied to the correct level.



If you've placed your **BP_EPWWB_Server_WorldManager** in the world, you should see a collision box indicating the size of the level or chunk.

You can adjust this size by selecting the **BP_EPWWB_Server_WorldManager**, going to the **Level Size** section in the **Details panel**, and modifying the **Level Chunk Size** property.
By default, the value is set to **6000, 6000, 6000**.





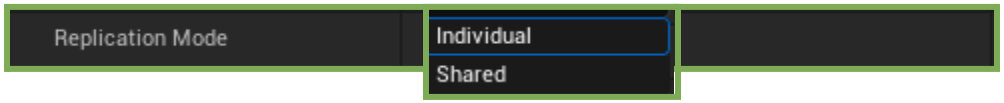
Then you can place any type of actor in your level.
For optimal results, it's recommended to keep all actors inside the collision box and avoid exceeding its boundaries.

Once your level design is complete, you can add your level to the **LevelsInfo** array of the biome of your choice.

Settings Overviews

Multiplayer

Replication Mode



Type: *Enumeration*

Defines how levels are replicated across clients.

- **Shared**: All clients share the same generated levels, meaning every player sees the level generation of all others.
- **Individual**: Each client loads their own levels independently, meaning every player sees their own level generation without affecting others. *(⚠️ Experimental)*

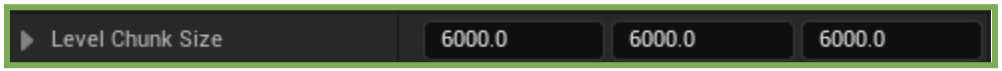
Hide Out Of Range Players ?

Type: *Boolean*

If enabled, players outside the currently loaded levels will be hidden.

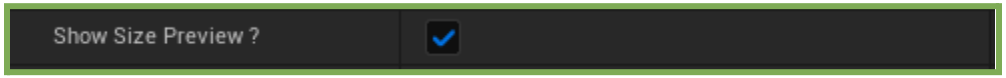
Levels Size

Level Chunk Size
Type: *Vector*



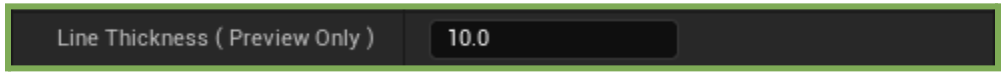
Defines the size of each streamed level chunk in world units (cm).
This value determines the dimensions of individual chunks used for level streaming.

Show Size Preview ?
Type: *Boolean*



If enabled, displays a visual representation of the level chunk size in the editor.
Useful for debugging level streaming boundaries.

Line Thickness (Preview Only)
Type: *Float*



Defines the thickness of the preview lines when displaying the level chunk size in the editor.
⚠️ ***Note:*** *This setting is only applicable if the Show Size Preview is set to True.*

Line Color (Preview Only)
Type: *Color*

Defines the color of the preview lines when displaying the level chunk size in the editor.
⚠️ ***Note:*** *This setting is only applicable if the Show Size Preview is set to True.*

World Generations

Coordinate Update Interval
Type: *Float*

Defines the time interval (in seconds) at which the player's coordinates are checked.

Timer Start Delay
Type: *Float*

Specifies the delay (in seconds) before the update timer begins.

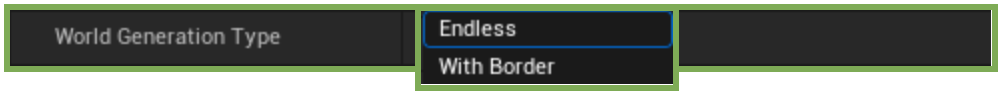
Block On Load Level?
Type: *Boolean*

When enabled, the screen freezes every time a level is loaded.

Block On Unload Level?
Type: *Boolean*

When enabled, the screen freezes every time a level is unloaded.

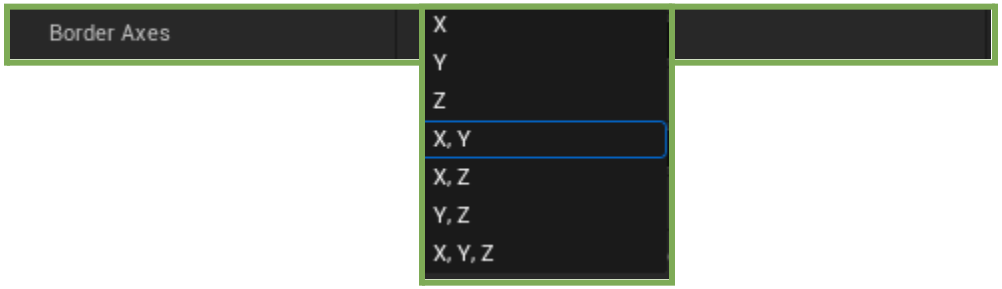
World Generation Type
Type: *Enumeration*



Defines the type of world generation used for the environment.

- **Endless:** The world expands infinitely as the player moves, creating a boundless environment.
- **With Border:** The world is generated with predefined borders.

Border Axes
Type: *Enumeration*



Define which axes the borders are applied to.

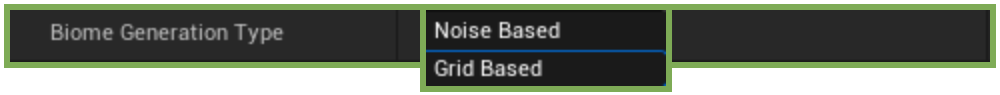
- ***X:** Border applied on the X axis only.*
- ***Y:** Border applied on the Y axis only.*
- ***Z:** Border applied on the Z axis only.*
- ***X, Y:** Borders applied on both the X and Y axes.*
- ***X, Z:** Borders applied on both the X and Z axes.*
- ***Y, Z:** Borders applied on both the Y and Z axes.*
- ***X, Y, Z:** Borders applied across all three axes.*

⚠ ***Note:** This setting is only applicable if the World Generation Type is set to With Border.*

Border Coords
Type: *Int Vector*

Defines the position of the world borders in terms of levels along each axis.
For example, if the values are set to X: 10, Y: 10, and Z: 2, it means the world borders are positioned 10 levels along the X axis, 10 levels along the Y axis, and 2 levels along the Z axis.
⚠ ***Note:** This setting is only applicable if the World Generation Type is set to With Border.*

Biome Generation Type
Type: *Enumeration*



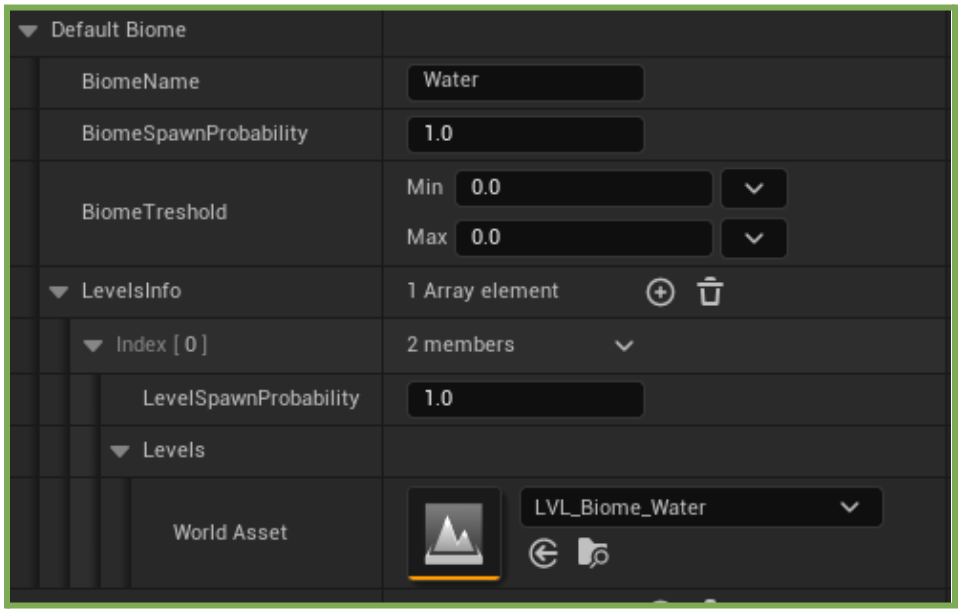
Specifies how biomes are generated in the world.

- **Noise Based:** Uses noise.
- **Grid Based:** Generates biomes in a structured grid pattern.

Biome Grid Size
Type: *Int Vector*

Defines the size of each grid cell, where each cell corresponds to a biome.
⚠️ ***Note:*** *This setting is only applicable if the Biome Generation Type is set to Grid Based.*

Default Biome
Type: *Structure*



Specifies the default biome that will be selected if no specific biome is found.

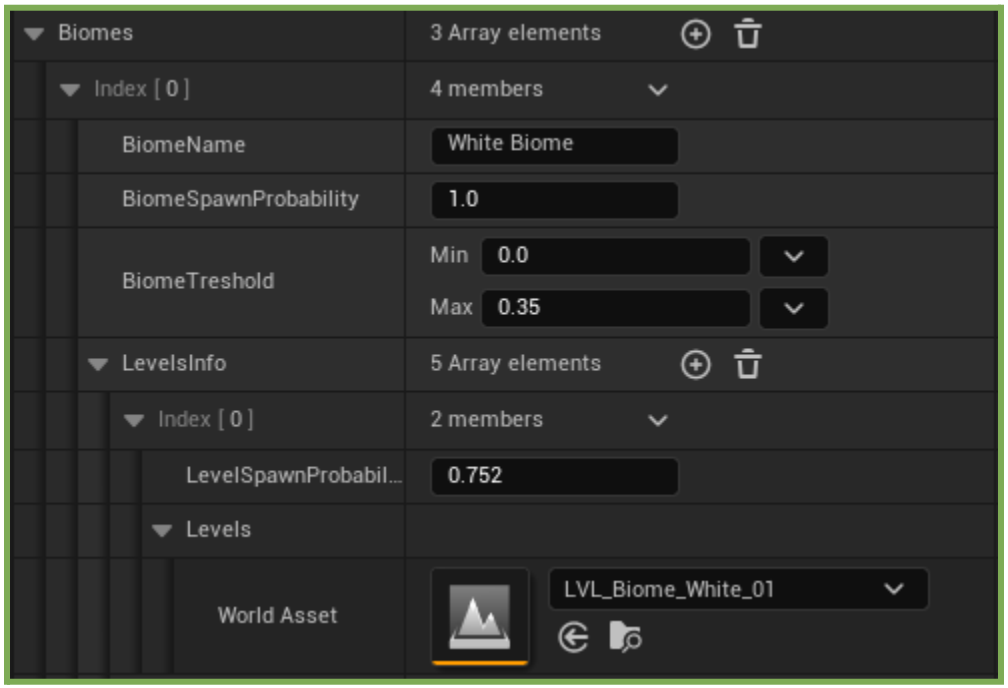
- **Biome Name:** Defines the name of the biome. This has no impact on generation and is used only for organizational purposes.
- **Biome Spawn Probability:** This value is not used for the default biome, as the default biome only spawns when no other biome is selected.
- **Biome Threshold:** This value is not used for the default biome.
- **Levels Info:** Add a list of levels to the biome, each with its own spawn probability and associated level.
  - **Level Spawn Probability:** Defines the

probability of spawning a specific level within the biome.

- **Levels:** Select which levels will be added to the biome.

Biomes

Type: *Structure*



Defines a list of biomes with their associated levels.

- **Biome Name:** Defines the name of the biome.
  This has no impact on generation and is used only for organizational purposes.

- **Biome Spawn Probability:** Defines the probability of this biome spawning in the world.

- **Biome Threshold:** Defines the noise value range required for the biome to spawn.
  - **Min**: Sets the minimum noise value required for the biome to spawn.
    The biome will only appear in areas where the noise value is greater than or equal to this threshold.

  - **Max**: Sets the maximum noise value at which the biome can spawn.
    The biome will only appear in areas where the noise value is less than or equal to this threshold.

  ⚠️ ***Note:*** *This setting is only applicable if the Biome Generation Type is set to Noise-Based.*

- **Levels Info:** Add a list of levels to the biome, each with its own spawn probability and associated level.
  - **Level Spawn Probability:** Defines the probability of spawning a specific level within the biome.

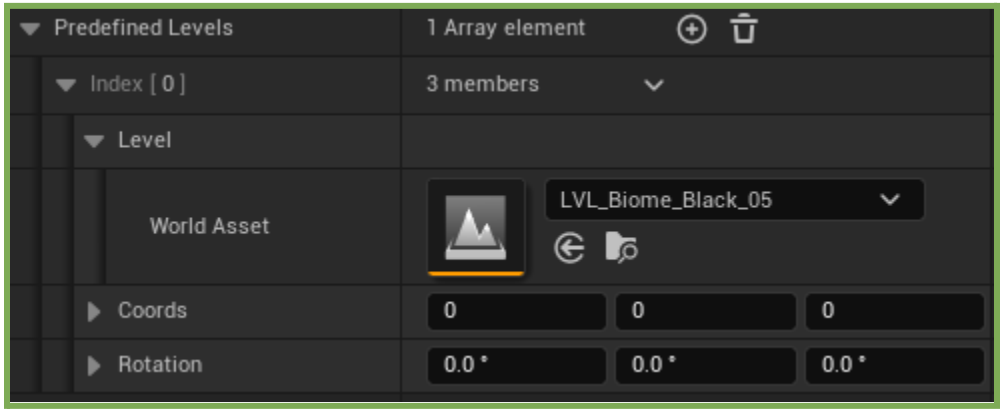
  - **Levels:** Select which levels will be added to the biome.

Predefined Levels

Type: *Structure*

Defines a list of predefined levels that will be placed at specific coordinates in the world.

- **Levels:** Specifies which levels will be added to the world.
- **Coords:** Defines the coordinates where the predefined level will be placed.
- **Rotation:** Specifies the rotation of the predefined level.



Levels Lifetime

Use Levels Lifetime ?
Type: *Boolean*

When the Use Levels Lifetime option is enabled, you can
define the lifetime of unused levels.
An unused level is a level that is loaded in memory but neither active nor visible to players.
When a level is unloaded, its coordinates and reference are stored in an array to allow for quick reloading without recreating the level.
If the level remains unused for a set period, it can be automatically removed to improve performance.

Days
Type: *Integer*

Number of days an unused level is retained in memory.
⚠️ ***Note:*** *This setting is only applicable if the Use Levels Lifetime is set to True.*

Hours
Type: *Integer*

Number of hours an unused level is retained in memory.
⚠️ ***Note:*** *This setting is only applicable if the Use Levels Lifetime is set to True.*

Minutes
Type: *Integer*

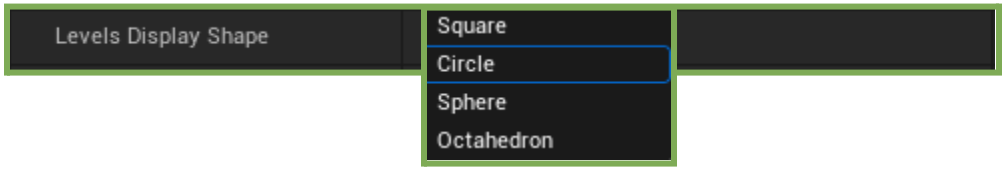
Number of minutes an unused level is retained in memory.
⚠️ ***Note:*** *This setting is only applicable if the Use Levels Lifetime is set to True.*

Seconds
Type: *Integer*

Number of seconds an unused level is retained in memory.
⚠️ ***Note:*** *This setting is only applicable if the Use Levels Lifetime is set to True.*

Levels Generations

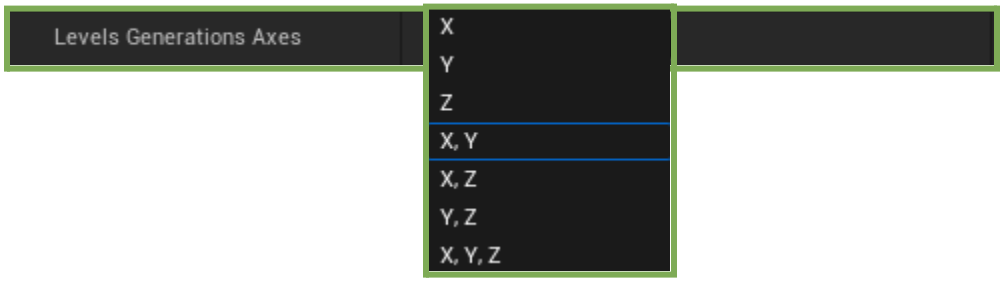
Levels Display Shape
Type: *Enumeration*



Defines the arrangement pattern of levels around the player.

- **Square:** Levels are arranged in a square pattern. If levels are also generated along the Z-axis, this forms a cube.
- **Circle:** Levels are arranged in a circular pattern. If levels are also generated along the Z-axis, this forms a cylinder.
- **Sphere:** Levels are arranged in a spherical pattern. A true sphere is only formed if levels are also generated along the Z-axis; otherwise, the pattern remains a circle.
- **Octahedron:** Levels are arranged in an octahedral pattern. A true octahedron is only formed if levels are also generated along the Z-axis; otherwise, the pattern remains a circle.

Levels Generations Axes
Type: *Enumeration*

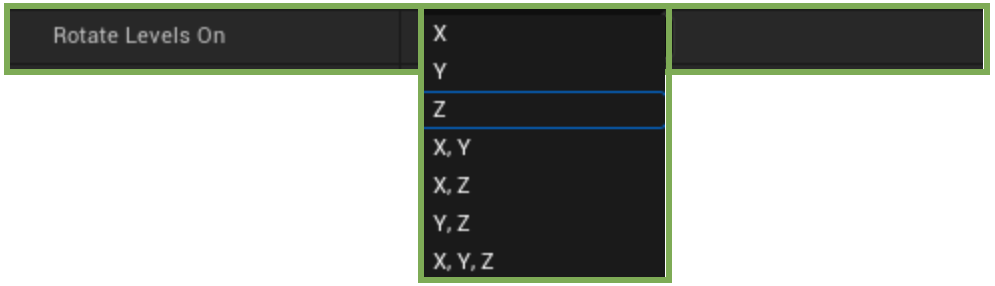


Defines the axes on which levels will be generated:

- **X:** Levels will be generated only along the X axis.
- **Y:** Levels will be generated only along the Y axis.
- **Z:** Levels will be generated only along the Z axis.
- **X, Y:** Levels will be generated along both the X and Y axes.
- **X, Z:** Levels will be generated along both the X and Z axes.
- **Y, Z:** Levels will be generated along both the Y and Z axes.
- **X, Y, Z:** Levels will be generated along all three axes

Levels Rotation

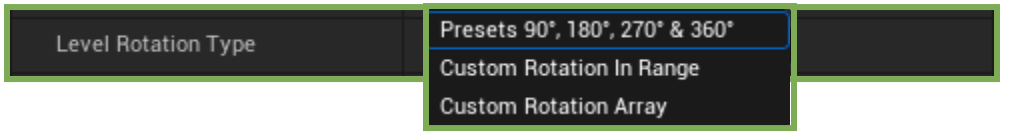
Rotate Levels On
Type : *Enumeration*



Defines the axes around which the level's rotation is applied.

- **X:** Rotation is applied only around the X-axis.
- **Y:** Rotation is applied only around the Y-axis.
- **Z:** Rotation is applied only around the Z-axis.
- **X, Y:** Rotation is applied around both the X and Y axes.
- **X, Z:** Rotation is applied around both the X and Z axes.
- **Y, Z:** Rotation is applied around both the Y and Z axes.
- **X, Y, Z:** Rotation is applied around all three axes.

Level Rotation Type
Type : *Enumeration*



Defines the method used for rotating the levels:

- **Presets 90°, 180°, 270° & 360°:** Levels are rotated by 90°, 180°, 270°, or 360°.

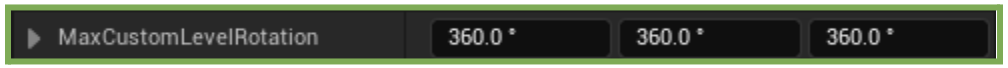
- **Custom Rotation In Range:** Defines the minimum and maximum values for the rotation of levels. For example, if Min Custom Level Rotation is set to X: 0.0, Y: 0.0, Z: 0.0, and Max Custom Level Rotation is set to X: 360.0, Y: 360.0, Z: 360.0, the levels will be rotated with random values between these minimum and maximum settings. For instance, a level could have a random rotation of X: 255.7, Y: 10.1, Z: 142.0.

- **Custom Rotation Array:** Defines specific rotation values within an array. This is useful for applying precise, predefined rotation values to levels.

Min Custom Level Rotation
Type: *Rotator*

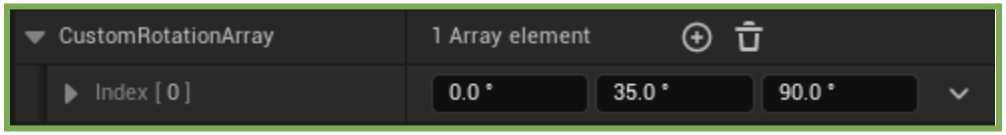
Defines the minimum values for the rotation of levels.
⚠️*Note: This setting is only applicable if the Level Rotation Type is set to Custom Rotation In Range.*

Max Custom Level Rotation
Type: *Rotator*



Defines the maximum values for the rotation of levels.
⚠️***Note:** This setting is only applicable if the Level Rotation Type is set to Custom Rotation In Range.*

Custom Rotation Array
Type: *Rotator*



Defines specific rotation values within an array.
⚠️***Note:** This setting is only applicable if the Level Rotation Type is set to Custom Rotation Array.*

Seed

Use Random Seed ?
Type: *Boolean*

Defines whether to generate a random seed each time a new world is created.

Seed
Type: *Integer*

Defines the world seed.
⚠️ ***Note:*** *This setting is only applicable if the Use Random Seed is set to False.*

Seed Offset
Type: *Int Vector*

Defines the offset applied to the base seed to introduce variation in the generation.

Noise

Noise Scale
Type: *Float*

Defines the scale of the noise used for biome generation. Higher values result in more zoomed-in noise, creating larger biome features. Lower values produce more detailed noise with smaller features.

Noise Amplitude
Type: *Float*

Defines the intensity of the variations generated by the noise.

Functions Library

Set Max Render Distance

Parameters:

- **Player Controller** (*Player Controller*): The player for whom the render distance will be set.
- **Max Render Distance** (*Int Vector*): The maximum render distance value.

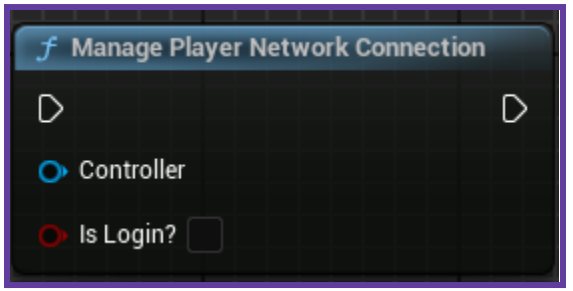
Defines the maximum render distance for a specific player, determining how far levels are rendered around them.

Manage Player Network Connection

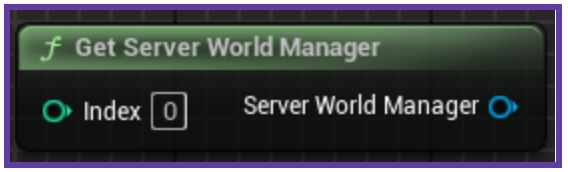
Parameters:

- **Controller** (*Controller*): The controller of the player.
- **Is Login ?** (*Boolean*):
  - True: The player is connecting to the server.
  - False: The player is disconnecting from the server.

Called from the **Game Mode**, this function handles player network connections.



Get Server World Manager



Parameters:

- **Index (*Integer*):** The index used to identify the target Server World Manager instance.

Return:

- **Server World Manager (*Server World Manager*):** Returns the Server World Manager associated with the given index.

Retrieves a specific Server World Manager instance based on the provided index.

Get Client World Manager

Parameters:

- **Index (*Integer*):** The index used to identify the target Client World Manager instance.

Return:

- **Client World Manager (*Client World Manager*):** Returns the Client World Manager associated with the given index.

Retrieves a specific Client World Manager instance based on the provided index.

Get Render Distance

Return:

- **Render Distance (*Int Vector*):** Returns the player's render distance.

This function returns the player's render distance, which determines how many chunks (Levels) are loaded and visible around the player.

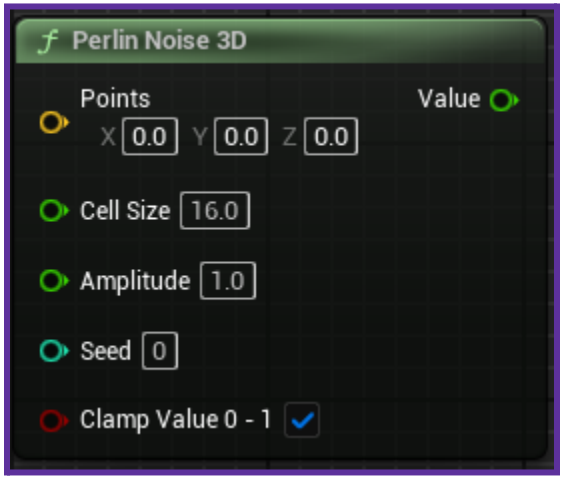
Get Current World Seed

Return:

- **Seed** (*Integer*): Returns the current world seed.

Retrieves the current world seed used for procedural world generation.

Perlin Noise 3D



Parameters:

- **Points** (*Vector*): A vector representing the coordinates (x, y, z) where the noise value will be evaluated.
- **Cell Size** (*Float*): Controls the scale of the noise. Larger values produce broader, smoother noise patterns, while smaller values result in more detailed, finer variations.
- **Amplitude** (*Float*): Determines the strength or intensity of the noise. Higher values amplify the noise output.
- **Seed** (*Integer*): An integer value used to initialize the random number generator, ensuring repeatable noise patterns for the same input.
- **Clamp Value 0 - 1** (*Boolean*): If enabled, clamps the final noise value between 0 and 1. Useful for keeping values within a normalized range.

Return:

- **Value** (*Float*): The calculated Perlin noise value at the specified point.

Generates 3D Perlin noise at a given point in space.