# Easy Monster Spawner

Professional Wave-Based Enemy Spawning System for Unreal Engine

v3.0.0 | UE 5.3 – 5.7 | C++ Plugin

# 🚀 How To Use

Easy Monster Spawner works in two modes: TriggerBox mode (classic — spawn when player enters an area) or Standalone mode (place on any actor, trigger from Blueprint/C++). Here's how to set up both.

## Option A — TriggerBox Mode (Classic)

1 **Enable the Plugin**

In Unreal Editor go to Edit → Plugins, find Easy Monster Spawner, enable it, and restart the editor.

2 **Place a TriggerBox**

From Place Actors → All Classes, drag a TriggerBox into your level. Scale it to cover the area where you want player entry to trigger spawns.

3 **Add the Component**

Select the TriggerBox, click +Add Component, search for "Monster Spawner" and add it.

4 **Set Monster Class**

In the Details panel under Classic Configuration, set Monster Class to your enemy Blueprint (e.g. BP_Zombie). Set Spawn Count, Spawn Pattern, and Spawn Radius.

5 **Play!**

Hit Play and walk into the TriggerBox. Monsters will spawn around it.

> 💡 Tip: Enable `Show Debug Visualization` in the Debug category to see spawn locations as cyan spheres in the editor viewport.

### Option B — Standalone Mode (NEW in v3.0)

**1** **Add to Any Actor**

Add the Monster Spawner component to any actor in your level — an empty Actor, a Blueprint, a character, anything.

**2** **Set Activation Mode**

In the Activation category, change Activation Mode to one of: `Manual`, `Timer Auto-Spawn`, or `Begin Play`.

**3** **Configure & Trigger**

Set up your monster class and spawn settings as normal. For Manual mode, call `TriggerSpawn()` from your Blueprint or C++ code when ready.

> ℹ️ If Activation Mode is set to TriggerBox but the owner actor is not a TriggerBox, the plugin automatically falls back to Manual mode and logs a warning.

### Using the Wave System

**1** **Enable Waves**

Check Use Wave System in the Wave System category.

**2** **Add Wave Entries**

Click + on the Waves array to add wave configurations. Each wave has its own Monster Class, Spawn Count, Pattern, Effects, Formation, and Clear Requirements.

**3** **Optional: Wave Clear**

In each wave's Clear Requirements, enable `Require Wave Clearance`. Set the kill percentage required before the next wave starts.

**4** **Optional: Loop**

Enable Loop Waves and set Difficulty Scale Per Loop for infinite scaling encounters.

## Blueprint Integration — Quick Example

```
// Trigger spawn from a gameplay event (e.g. button press)
Event AnyEvent
  → Get Component by Class (Monster Spawner v3.0)
    → TriggerSpawn()

// Check if all monsters are dead
Event Tick
  → Get Component by Class (Monster Spawner v3.0)
    → GetAliveMonsterCount()
      → Branch (Count == 0)
          → True: Open Door / Give Reward
```

## ✨ Feature Overview

NEW v3.0

### 🔓 Standalone Mode

No TriggerBox needed. 4 activation modes: TriggerBox, Manual, Timer, BeginPlay.

NEW v3.0

### 🎢 Multi-Class Waves

Mix enemy types with weighted random selection per wave or classic mode.

NEW v3.0

### ♻️ Object Pooling

Reuse actors for performance. Pre-allocate, auto-expand, return to pool.

NEW v3.0

### 🌐 Multiplayer

Server-authoritative spawning with replicated state sync.

NEW v3.0

### 👁️ Spawn Conditions

Line of sight, "not looking" horror mode, min/max player distance.

NEW v3.0

### 🔊 MetaSound + Attenuation

USoundBase support (SoundCue, MetaSound, SoundWave) + 3D attenuation.

v2.5

### 💥 Spawn Effects

Niagara VFX + SFX for pre-spawn and post-spawn with configurable delay.

v2.5

### ⚔️ RTS Formations

8 formation types: Wedge, Line, Column, Square, Staggered, Phalanx, Circle, Scatter.

v2.5

### 💀 Wave Clear

Require kill percentage before next wave. Timeout with force-proceed option.

v2.5

### 🌀 New Patterns

Line, Arc, V-Shape, Spiral spawn patterns.

v2.0

### 🌊 Multi-Wave System

Sequential waves with per-wave config, delay, difficulty scaling, loop.

v1.0

### 🧠 Smart Spawn

Ground trace, NavMesh validation, overlap avoidance, ground alignment.

## Activation Modes

Set via the `Activation Mode` property in the Activation category.

| Mode | How It Works | Owner Requirement |
|---|---|---|
| `TriggerBox` | Spawns when a player-controlled character overlaps the TriggerBox. Classic v1.0 behavior. | Must be ATriggerBox |
| `Manual` | Only spawns when you call `TriggerSpawn()` from Blueprint or C++. | Any actor |
| `TimerBased` | Automatically spawns on a repeating interval set by `AutoSpawnInterval` (seconds). | Any actor |
| `BeginPlay` | Spawns immediately when the game starts. | Any actor |

| Property | Type | Default | Description |
|---|---|---|---|
| `ActivationMode` | ESpawnActivationMode | TriggerBox | How this spawner is activated |
| `AutoSpawnInterval` | Float | 30.0 | Seconds between auto-spawns (TimerBased only, 0.1–300) |

## ⚙️ Classic Configuration

Used when `Use Wave System` is disabled. Single-trigger spawning.

| Property | Type | Default | Description |
|---|---|---|---|
| `MonsterClass` | TSubclassOf<AActor> | None | Actor class to spawn (e.g. BP_Zombie) |
| `ClassicMonsterClasses` | TArray<FMonsterClassEntry> | [] | Multi-class list with weights. If populated, overrides MonsterClass. v3.0 |
| `SpawnCount` | Int32 | 5 | How many monsters to spawn (1–100) |
| `SpawnPattern` | ESpawnPattern | Circle | Spawn placement pattern |
| `SpawnRadius` | Float | 500 | Spawn area radius in cm (100–5000) |
| `SpawnDelay` | Float | 0.2 | Seconds between each monster spawn (0 = all at once) |
| `DeactivationDelay` | Float | 0.0 | Seconds before trigger deactivates after spawn starts |
| `bFaceTriggerCenter` | Bool | false | Rotate spawned monsters to face the trigger/owner center |
| `AdditionalTagsForSpawnedMonsters` | TArray<FName> | [] | Tags added to every spawned monster |

## General Settings (both modes)

| Property | Type | Default | Description |
|---|---|---|---|
| `bOneTimeOnly` | Bool | true | Trigger works only once? |
| `SpawnHeightOffset` | Float | 50 | Z-offset above ground for spawn location (0–500 cm) |

## Wave System

| Property | Type | Default | Description |
|---|---|---|---|
| `bUseWaveSystem` | Bool | false | Enable multi-wave spawning |
| `Waves` | TArray<FWaveConfiguration> | [] | Array of wave configs |
| `bLoopWaves` | Bool | false | Restart from wave 0 after last wave completes |
| `DifficultyScalePerLoop` | Float | 1.2 | Multiplier for SpawnCount each loop (1.0–3.0) |

## FWaveConfiguration (per wave)

| Property | Type | Description |
|---|---|---|
| `MonsterClass` | TSubclassOf<AActor> | Single monster class for this wave |
| `MonsterClasses` | TArray<FMonsterClassEntry> | Multi-class list (overrides MonsterClass if populated) v3.0 |
| `SpawnCount` | Int32 | How many to spawn (1–100) |
| `SpawnPattern` | ESpawnPattern | Spawn placement pattern |
| `SpawnRadius` | Float | Radius in cm (100–5000) |
| `SpawnDelay` | Float | Delay between each monster (0–10s) |
| `DelayBeforeNextWave` | Float | Seconds after this wave before next starts (0–60) |
| `bFaceTriggerCenter` | Bool | Rotate to face trigger/owner center |
| `MonsterTags` | TArray<FName> | Tags for this wave's monsters |
| `bUseCustomEffects` | Bool | Override global effects for this wave |
| `CustomSpawnEffects` | FSpawnEffectConfig | Per-wave effects (if above is true) |
| `bUseFormation` | Bool | Use formation spawning for this wave |
| `FormationConfig` | FFormationConfig | Per-wave formation settings |
| `ClearRequirement` | FWaveClearRequirement | Kill requirement before next wave |

## 🐉 Multi-Class Waves v3.0

Mix different enemy types in a single wave using weighted random selection.

### FMonsterClassEntry

| Property | Type | Default | Description |
|---|---|---|---|
| `MonsterClass` | TSubclassOf<AActor> | None | The monster class |
| `SpawnWeight` | Int32 | 1 | Relative weight (higher = more likely, 1–100) |
| `MonsterTags` | TArray<FName> | [] | Tags unique to this monster type |

### Example Setup

```
// 70% Skeletons, 20% Trolls, 10% Boss
ClassicMonsterClasses:
  [0] MonsterClass: BP_Skeleton, SpawnWeight: 7
  [1] MonsterClass: BP_Troll,    SpawnWeight: 2
  [2] MonsterClass: BP_MiniBoss, SpawnWeight: 1
```

> ℹ️ If `MonsterClasses` (wave) or `ClassicMonsterClasses` (classic) is populated, it overrides the single `MonsterClass` field. You can use both in the same project — single class for simple waves, multi-class for variety waves.

## 🎯 Spawn Patterns

9 patterns available via the `ESpawnPattern` enum:

`Circle` `Grid` `Random` `SpawnPoints` `Line` `Arc` `VShape` `Spiral` `Formation`

### 🔵 Circle

Evenly spaced around center at SpawnRadius distance. Great for surrounding the player.

### 📐 Grid

NxN grid within SpawnRadius. Perfect for organized formations.

### 🎲 Random

Random positions within SpawnRadius. Unpredictable encounters.

### 📍 Spawn Points

Uses child SceneComponents of the owner as custom positions. Falls back to Circle if none found.

### ➖ Line

Straight line across SpawnRadius×2 length.

### 🌙 Arc

180° arc at SpawnRadius distance.

### ✌️ V-Shape

V/arrow formation with leader at front.

### 🌀 Spiral

Outward spiral with 2 full rotations.

### ⚔️ Formation

Uses the RTS Formation System (see below).

# ⚔️ RTS Formation System

8 formation types via `EFormationType`:

`Wedge` `Line` `Column` `Square` `Staggered` `Phalanx` `Circle` `Scatter`

## FFormationConfig

| Property | Type | Default | Description |
|---|---|---|---|
| `FormationType` | EFormationType | Wedge | Which formation to use |
| `UnitSpacing` | Float | 150 | Distance between units (50–500 cm) |
| `FormationFacingAngle` | Float | 0 | Rotation offset from owner's forward (-180 to 180°) |
| `bUnitsUseFacingDirection` | Bool | true | Units face the formation direction |
| `FormationDepth` | Int32 | 3 | Rows for multi-row formations (1–10) |
| `WedgeAngle` | Float | 45 | Wedge opening angle (15–90°) |
| `RandomOffset` | Float | 0 | Random jitter for organic look (0–100 cm) |

## Spawn Effects

Enable with `bUseSpawnEffects`. Configure globally or per-wave.

### FSpawnEffectConfig

| Property | Type | Description |
|---|---|---|
| `PreSpawnVFX` | TSoftObjectPtr<UNiagaraSystem> | Niagara effect before monster appears |
| `PostSpawnVFX` | TSoftObjectPtr<UNiagaraSystem> | Niagara effect after monster spawns |
| `PreSpawnSFX` | TSoftObjectPtr<USoundBase> | Sound before spawn (SoundCue, MetaSound, SoundWave) v3.0 |
| `PostSpawnSFX` | TSoftObjectPtr<USoundBase> | Sound after spawn v3.0 |
| `SoundAttenuation` | TSoftObjectPtr<USoundAttenuation> | 3D audio falloff settings v3.0 |
| `PreSpawnDelay` | Float | Seconds between pre-effect and actual spawn (0–5) |
| `EffectScale` | Float | VFX scale multiplier (0.1–10) |
| `bAttachPostEffectToMonster` | Bool | Attach post-spawn VFX to the actor |

## 👁️ Spawn Conditions v3.0

Control where and when monsters can spawn relative to the player.

### FSpawnConditions

| Property | Type | Default | Description |
|---|---|---|---|
| `bRequireLineOfSight` | Bool | false | Only spawn at locations visible to any player (not blocked by geometry) |
| `bSpawnOnlyWhenPlayerNotLooking` | Bool | false | Only spawn when no player is looking at the location (horror mode!) |
| `PlayerLookAngleThreshold` | Float | 60 | FOV half-angle to consider "looking at" (10–180°) |
| `MinDistanceFromPlayer` | Float | 0 | Minimum distance from any player (0 = no limit) |
| `MaxDistanceFromPlayer` | Float | 0 | Maximum distance from any player (0 = no limit) |

> 💡 Horror game tip: Set `bSpawnOnlyWhenPlayerNotLooking = true` + `MinDistanceFromPlayer = 500` for enemies that appear behind you when you're not looking!

## 💀 Wave Clear Requirements

### FWaveClearRequirement

| Property | Type | Default | Description |
|---|---|---|---|
| `bRequireWaveClearance` | Bool | false | Must kill monsters before next wave? |
| `RequiredKillPercentage` | Float | 100 | % of monsters that must die (0–100) |
| `MaxWaitTime` | Float | 0 | Timeout in seconds (0 = infinite) |
| `bForceNextWaveOnTimeout` | Bool | true | Force next wave when timeout expires |

## ♻️ Object Pooling v3.0

Reuse actor instances instead of Spawn/Destroy for better performance with large waves.

| Property | Type | Default | Description |
|---|---|---|---|
| `bUseObjectPooling` | Bool | false | Enable pooling |
| `InitialPoolSize` | Int32 | 10 | Pre-spawned hidden actors at BeginPlay (1–200) |
| `MaxPoolSize` | Int32 | 50 | Maximum pool capacity (0 = unlimited) |
| `bAutoExpandPool` | Bool | true | Create new actors when pool is empty |

### How Pooling Works

1. At BeginPlay, the pool pre-spawns `InitialPoolSize` actors hidden at (0,0,-10000).
2. When spawning, the system first tries to reuse a pooled actor of the requested class.
3. If pool is empty and `bAutoExpandPool` is true, a new actor is spawned (up to `MaxPoolSize`).
4. If pool is exhausted and can't expand, falls back to standard `SpawnActor`.
5. Call `ReturnToPool(Monster)` instead of `Destroy` to recycle the actor.

### Pool Blueprint Functions

| Function | Description |
|---|---|
| `ReturnToPool(AActor*)` | Deactivate and return an actor to the pool |
| `DestroyPool()` | Destroy all pooled actors (available + in use) |
| `GetPoolAvailableCount()` | Number of ready-to-use actors in pool |
| `GetPoolTotalSize()` | Total pool size (available + in use) |

## 🌐 Multiplayer Replication v3.0

| Property | Type | Default | Description |
|---|---|---|---|
| `bEnableReplication` | Bool | false | Enable server-authoritative spawning |

When enabled:

- Client calls to `TriggerSpawn()` are routed to the server via `Server_TriggerSpawn` RPC.
- `bHasTriggered`, `bSpawnInProgress`, `MonstersSpawned`, and `CurrentWaveIndex` replicate to all clients.
- Spawned actors follow standard UE replication (set your monster BP to replicate separately).

## 🧠 Smart Spawn System

| Property | Type | Default | Description |
|---|---|---|---|
| `bUseGroundTrace` | Bool | true | Line trace to find ground surface |
| `GroundTraceDistance` | Float | 1000 | Max downward trace distance (100–5000 cm) |
| `bUseNavMesh` | Bool | true | Validate spawn on NavMesh (AI can navigate) |
| `NavMeshQueryExtent` | Float | 500 | NavMesh search radius (100–1000 cm) |
| `bAvoidOverlap` | Bool | true | Prevent spawning on top of existing actors |
| `MinSpawnDistance` | Float | 100 | Minimum distance between spawned monsters (50–500 cm) |
| `MaxSpawnAttempts` | Int32 | 10 | Retries per monster to find valid location (1–50) |
| `bAlignToGround` | Bool | false | Align rotation to ground normal (for slopes) |

## 📘 Blueprint Functions

| Function | Type | Description |
|---|---|---|
| `TriggerSpawn()` | Callable | Start spawning (works in all modes) |
| `HasBeenTriggered()` | Pure | Returns true if spawner was already triggered |
| `ResetSpawner()` | Callable | Reset all state, re-enable trigger |
| `GetAliveMonsterCount()` | Pure | Number of alive monsters from this spawner |
| `GetSpawnedMonsters()` | Pure | Array of all alive spawned actors |
| `GetSpawnCenter()` | Pure | World location of spawn center (trigger or owner) v3.0 |
| `GetCurrentWaveIndex()` | Pure | Current wave index (0-based, -1 if not started) |
| `GetTotalWaves()` | Pure | Number of configured waves |
| `IsWaveInProgress()` | Pure | True if currently spawning a wave |
| `SkipToNextWave()` | Callable | Stop current wave, proceed to next |
| `GetCurrentWaveKillPercentage()` | Pure | % of current wave monsters killed |
| `IsCurrentWaveCleared()` | Pure | True if kill requirement met |
| `IsWaitingForWaveClear()` | Pure | True if waiting for player to clear wave |
| `ForceClearWave()` | Callable | Force-proceed to next wave |
| `ReturnToPool(AActor*)` | Callable | Return monster to object pool v3.0 |
| `DestroyPool()` | Callable | Destroy all pooled actors v3.0 |
| `GetPoolAvailableCount()` | Pure | Available pool actors v3.0 |

| | | |
|---|---|---|
| `GetPoolTotalSize()` | Pure | Total pool size v3.0 |

## Blueprint Events (Delegates)

| Event | Parameters | Description |
|---|---|---|
| `OnWaveStarted` | int32 WaveIndex | Fired when a wave begins spawning |
| `OnWaveCompleted` | int32 WaveIndex, int32 MonstersSpawned | Fired when all monsters in a wave are spawned |
| `OnAllWavesCompleted` | — | Fired when the last wave finishes |
| `OnWaveCleared` | int32 WaveIndex, float ClearTime | Fired when kill requirement is met |
| `OnMonsterSpawned` | AActor* Monster, int32 SpawnIndex | Fired for each individual monster spawn |
| `OnMonsterDied` | AActor* DeadMonster | Fired when a spawned monster is destroyed |
| `OnMonsterReturnedToPool` | AActor* PooledMonster | Fired when a monster is returned to pool v3.0 |
| `OnSpawnFailed` | int32 MonsterIndex, FVector Location | Fired when a spawn fails (no valid location) v3.0 |

## 🔍 Debug & Visualization

| Property | Type | Default | Description |
|---|---|---|---|
| `bShowDebugVisualization` | Bool | true | Draw spawn radius + preview points in editor |
| `bShowSpawnValidation` | Bool | false | Draw green/red spheres during runtime for valid/invalid spawn attempts |

All log messages use the `LogMonsterSpawner` category. Filter in Output Log with: `LogMonsterSpawner`

## 🔧 Troubleshooting

### Monsters Don't Spawn

- Check Monster Class is set (not None)
- For TriggerBox mode: owner must be an ATriggerBox
- For Standalone: check Activation Mode is not TriggerBox
- Check Output Log → filter `LogMonsterSpawner`
- If using multi-class: at least one entry must have a valid class

### Monsters Spawn Underground or Floating

- Adjust SpawnHeightOffset (default 50 cm)
- Enable Use Ground Trace
- Increase GroundTraceDistance if terrain is uneven

### Pool Not Working

- Ensure bUseObjectPooling is checked
- Check log for "No class for pre-allocation" — set MonsterClass first
- Call `ReturnToPool()` instead of `Destroy()` on your monsters

### Multiplayer Issues

- Enable bEnableReplication
- Ensure your monster Blueprints also replicate (Actor → Replicates = true)
- Spawning only happens on the server — state replicates automatically

> ⚠️ Performance: Spawning 50+ monsters at once (SpawnDelay=0) can cause frame drops. Use a delay of 0.1–0.3s, or enable Object Pooling for large waves.

Easy Monster Spawner v3.0.0 — by Erak_Dev

Unreal Engine 5.3 – 5.7 | C++ Plugin | Marketplace Standard EULA

adrenalinegames.pl