EASY CROSSHAIR PLUGIN

Introduction

Easy Crosshair Plugin is a plugin developed to accelerate and simplify the crosshair design workflow within Unreal Engine. With its built-in editor, it allows for quick and efficient creation of crosshairs.

The plugin primarily includes a **Crosshair Editor**, **Crosshair System**, and a variety of **Textures**. These components can be used to design custom crosshairs and seamlessly integrate them with game logic.

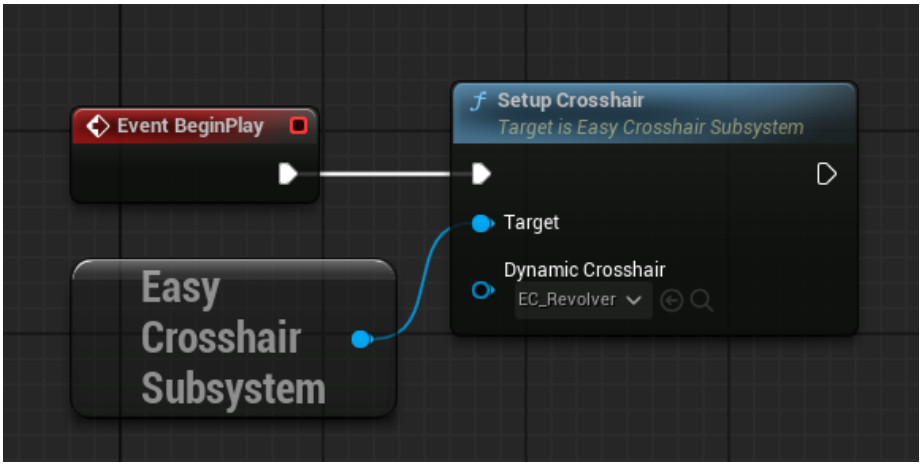
The plugin comes with **518 textures**, including Crosshairs, Kill Indicators, Armor Indicators, and Hit Markers. It provides a wide range of modular crosshair elements, enabling users to create their own designs. These elements can be freely combined using a **layer system** to achieve the desired visual style

Textures

The textures are located under the */Textures* directory and are organized into four categories: **ArmorIndicators**, **Crosshairs**, **Hitmarkers**, and **KillIndicators**.
Each texture has been imported at a resolution of **256x256**, ensuring clarity even at high resolutions.
For a crispy appearance, the **Maximum Texture Size** parameter in the **Texture** settings can be adjusted.
The recommended minimum value for this parameter is **128x128**.

Setup

There is no need to create a custom Widget to use the **Easy Crosshair System**, as the plugin handles this process internally. All you need to do is call the SetupCrosshair method provided by the **EasyCrosshairSubsystem** at the point in your game code where you want the crosshair to appear. This method can be called from either **Blueprint** or **C++**.

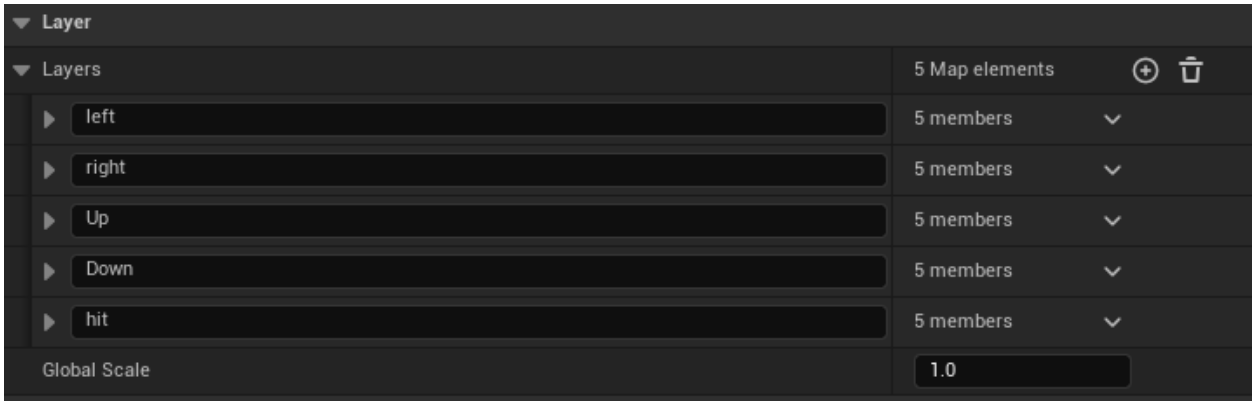


The SetupCrosshair method requires an **EasyCrosshair Asset** as input. An EasyCrosshair asset is a specialized Unreal Engine asset that holds a crosshair design and can be edited using the plugin's built-in editor. It is recommended to create separate EasyCrosshair assets for different weapons (e.g., EC_M4, EC_Shotgun).

Once set up, the crosshair will be automatically displayed at the appropriate position in the center of the screen. You can later remove it using the RemoveCrosshair method or play animations using the RunAnimation method—both accessible via the **EasyCrosshairSubsystem**.

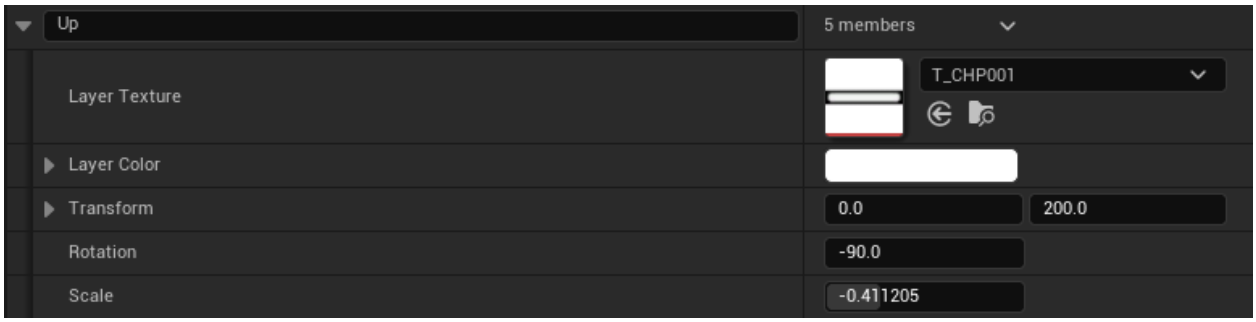
Layer System

The **Layer System** is the core component of our Crosshair Editor. It allows you to customize and fine-tune each individual element that makes up your crosshair—such as the **inner layer**, **outer layer**, **kill indicator**, and other modular parts. Through this system, you can precisely control the **position**, **color**, **size**, **rotation**, and **shape** of each layer, giving you complete creative freedom to design the exact crosshair look you envision.



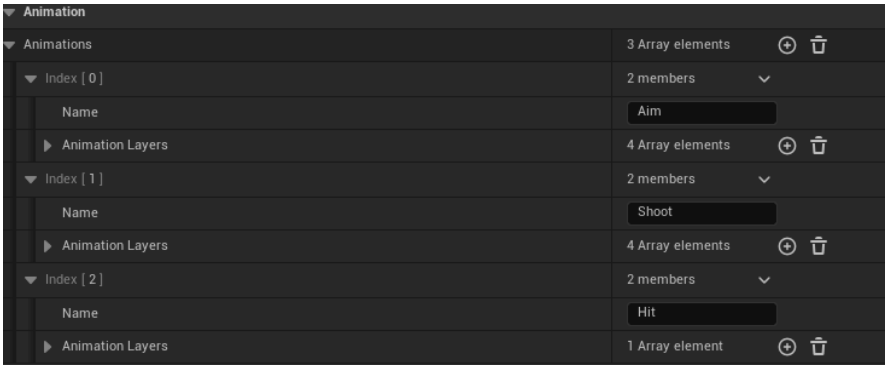
Each layer operates independently and can be stacked or combined with others to create complex visual effects. For example, you can create a multi-layered crosshair with glowing

outlines, animated indicators, or color-changing hit markers. The editor provides real-time **preview functionality**, allowing you to instantly see how changes affect the overall design, which greatly speeds up iteration and prototyping.

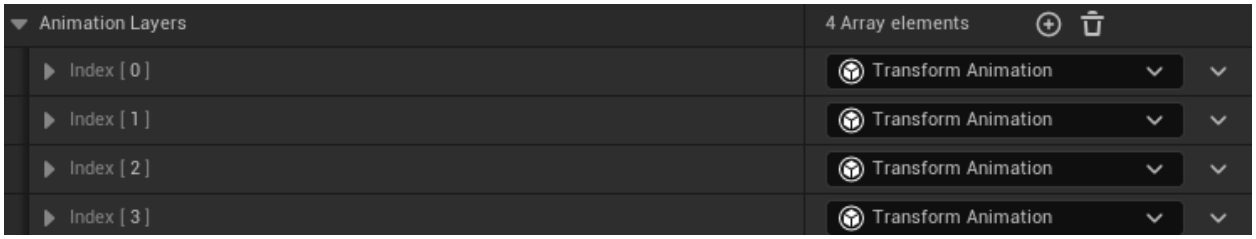


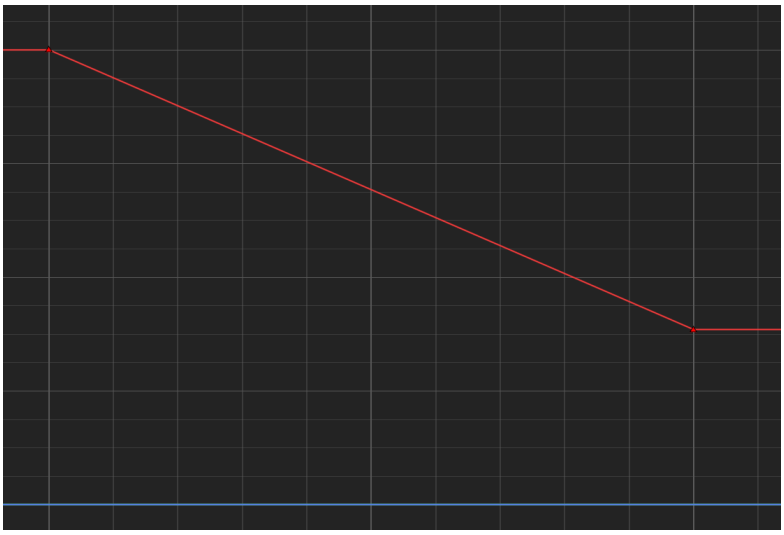
Additionally, layers can be grouped, reordered, and toggled on or off, giving you flexibility in organizing your design. This system is particularly useful when designing crosshairs for different gameplay scenarios, such as scoped modes, low-health indicators, or dynamic hit feedback. Whether you're creating a minimal competitive reticle or a stylized HUD element for an RPG, the Layer System serves as a powerful tool to bring your ideas to life directly within the editor.

Animations

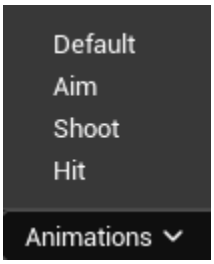


Animations allow you to bring your crosshair designs to life by adding dynamic, visually engaging effects. Each animation is composed of one or more **Animation Layers**, which define how specific properties of your crosshair elements change over time. Animation layers can be of two main types: **TransformAnimation** (modifying position, scale, rotation) and **ColorAnimation** (modifying color and opacity).



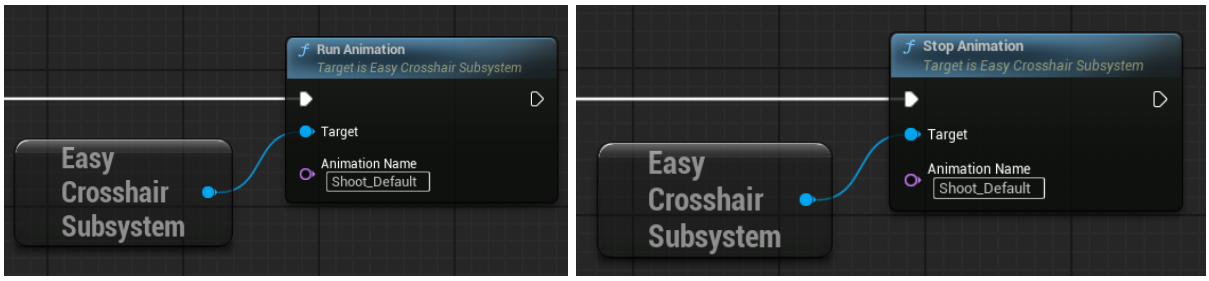


After creating an animation, you can fine-tune its behavior using **curves** and a variety of **tweakable options**, such as loop modes, delays, playback speed, and easing functions. These tools provide precise control over the timing and feel of each animation. For instance, you can create a smooth pulsing effect when aiming, a sharp recoil animation when shooting, or a color flash when an enemy is hit.



Animations can be previewed directly within the editor through the **Animations Panel**, located in the lower-left section of the editor interface. This allows you to test and iterate on your animation designs **without needing to run the game or write any code**, significantly speeding up your workflow. Real-time previewing makes it easy to adjust curves and layer parameters until the animation matches your vision.

Every animation must be given a unique **name**, as this identifier is used later in the game code to **play**, **stop**, or **trigger** the animation via the RunAnimation method in the EasyCrosshairSubsystem. By naming and organizing your animations properly, you can build a robust animation library tailored for different weapons, gameplay states, or user preferences.



Overall, the animation system is designed to be both artist-friendly and technically powerful, enabling you to add polish and responsiveness to your crosshair system with minimal effort.