HITWICKET ASSIGNMENT DOCUMENTATION

Problem Statement 01: Create a Cel Shader using Unity's Shader graph or Shader Programming for the 3D model.

Purpose of the Shader & Objective:

The objective of this shader is to create a stylized Cel (Toon) shading model for the provided 3D character while maintaining visual clarity, lighting readability, and controllable artistic parameters.

Unlike physically based shading, this shader can control lighting into distinct bands to achieve a stylized look while preserving form through controlled diffuse, specular, and fresnel effects.

NORMAL MODEL CEL SHADED MODEL

Properties That Drive the Cel Shader:

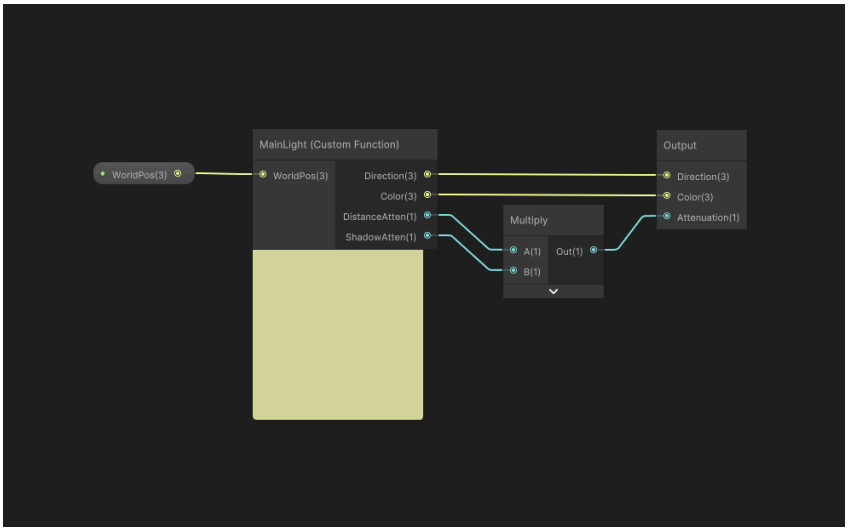
1. **MainTexture:** Base albedo texture of the character.
2. **Color:** Used for jersey color customization without modifying texture assets.
3. **Ambient Strength:** Controls how much indirect lighting influences the model. Prevents pure black shadows and preserves detail in unlit regions.
4. **Specular Color:** Defines the highlight color for stylized specular reflection.
5. **Smoothness:** Controls specular highlight sharpness.
6. **Fresnel Color:** Defines rim lighting color based on view angle.
7. **Fresnel Size:** Controls thickness of the rim effect.
8. **Lighting Cutoff**: Controls the threshold where light transitions into shadow.
9. **Light Falloff Amount:** Softens or sharpens the transition between lit and shadow bands.

FUNCTION OF THE SHADER

To get a proper Cel shading effect, we need to have the following components in the shader graph. I have given the explanation of those below:

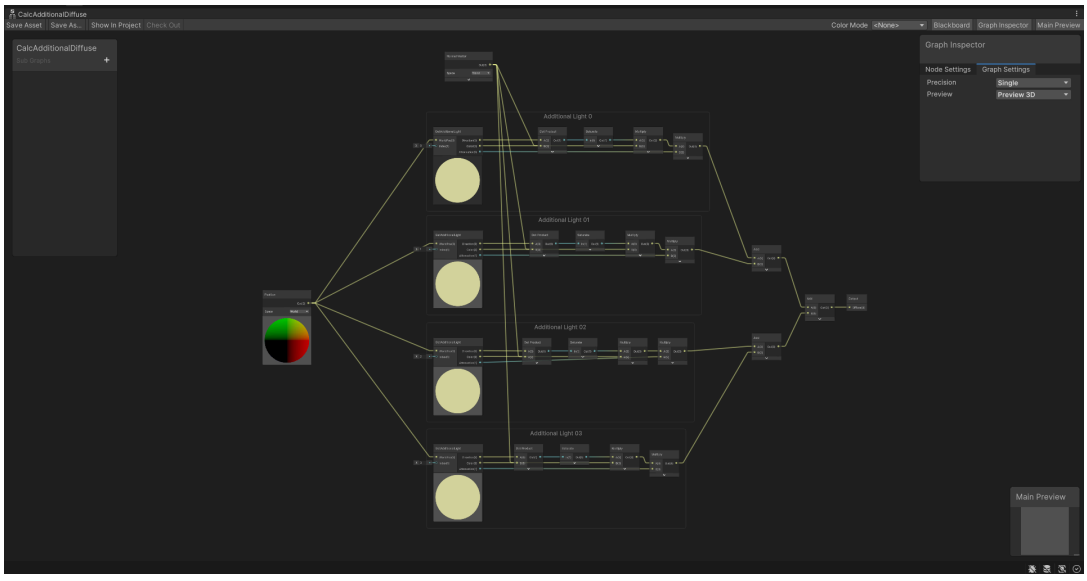
1. Function of GetMainLight & GetAdditionalLight: **GetMainLight()** is a custom HLSL shader function that fetches directional light data from Unity's URP lighting system.It provides Light direction,Light color, Attenuation, Shadow data

GetAdditionalLight() Retrieves additional lights affecting the object (point, spot). Used inside a loop to accumulate multiple light contributions. Ensures the cel shader remains compatible with dynamic lighting setups.

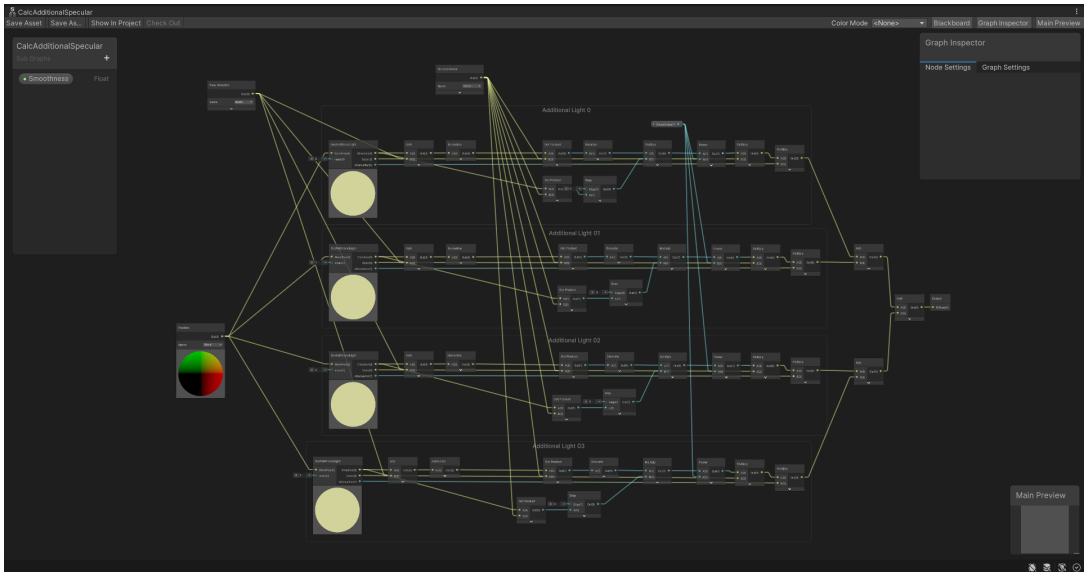


This function allows the shader to calculate primary N·L lighting using scene light information.

2. CalcAdditionalDiffuse: Calculates diffuse lighting contribution from additional lights. It Computes N·L, Applies attenuation, Applies stylized cutoff. This keeps additional lights consistent with the cel shading effect

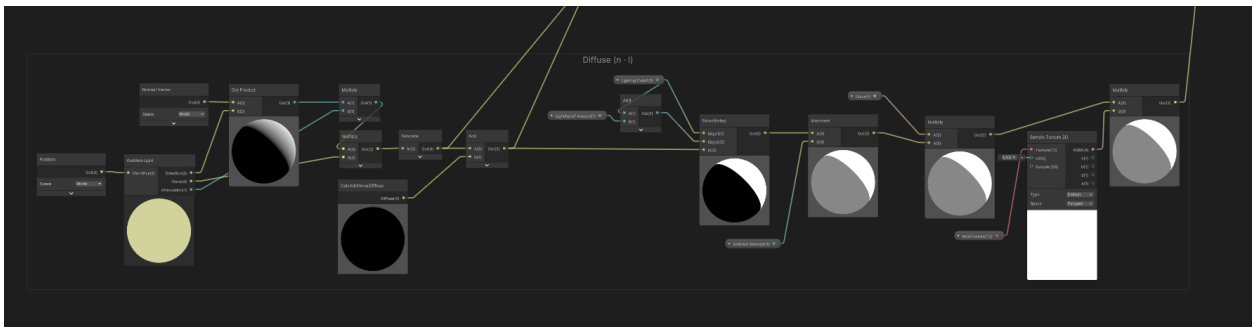


3. CalcAdditionalSpecular: Computes stylized specular highlights for additional lights.Uses: Half vector, Dot product. Specular is shaped and thresholded to match the toon aesthetic rather than realistic effect.

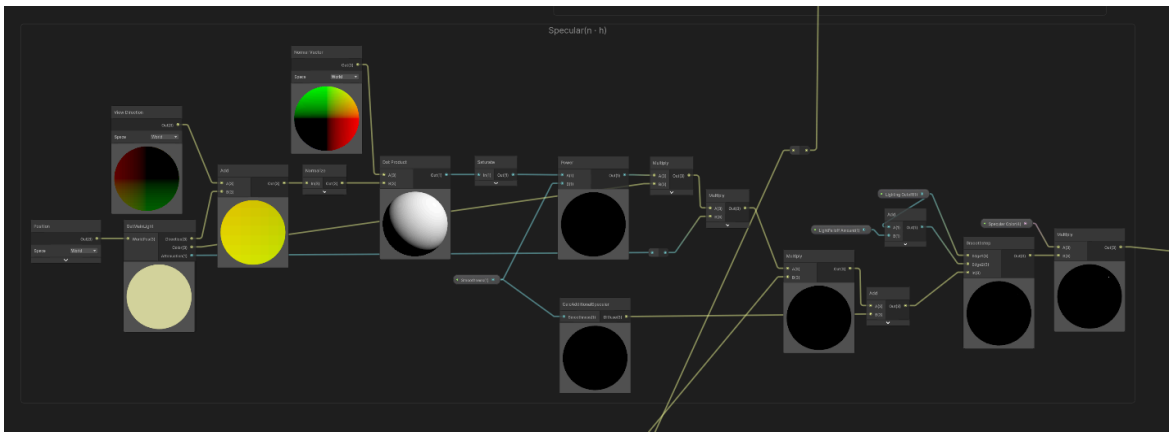


With these Custom Shader node components. We can achieve Diffuse Effect, Specular Effect and Fresnel Effect by combining them with ADD NODE. For the unlit material we get the Cel shader effect.

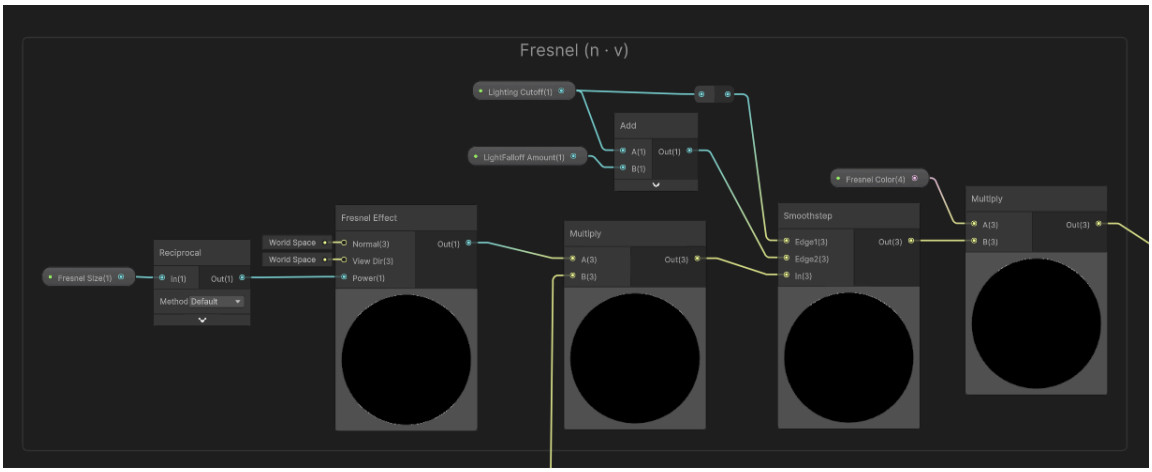
Diffuse (N · L): The diffuse section calculates how much the surface is facing the main light using the dot product between the world-space normal and light direction (N·L). This value represents the base light intensity on the surface. It is then adjusted using the Lighting Cutoff and Light Falloff parameters to control where the shadow begins and how sharp the transition appears. A smoothstep function is applied to convert the smooth gradient into distinct light bands, producing the cel-shaded look. Ambient strength is blended in to prevent pure black shadows, and the final result is multiplied with the base texture and color tint to produce stylized lit and shadow regions



.**Specular (N · H):** The specular section generates stylized highlights using the half-vector method. The light direction and view direction are combined and normalized to compute a half-vector, which is then compared with the surface normal using a dot product (N·H). This determines highlight intensity. The result is sharpened using a power function controlled by the Smoothness parameter, allowing tight or broad highlights. A Smoothstep operation thresholds the highlight to create hard-edged, toon-style reflections instead of realistic glossy shading. The highlight is tinted using the Specular Color and combined with contributions from additional lights to maintain compatibility with multiple light sources.

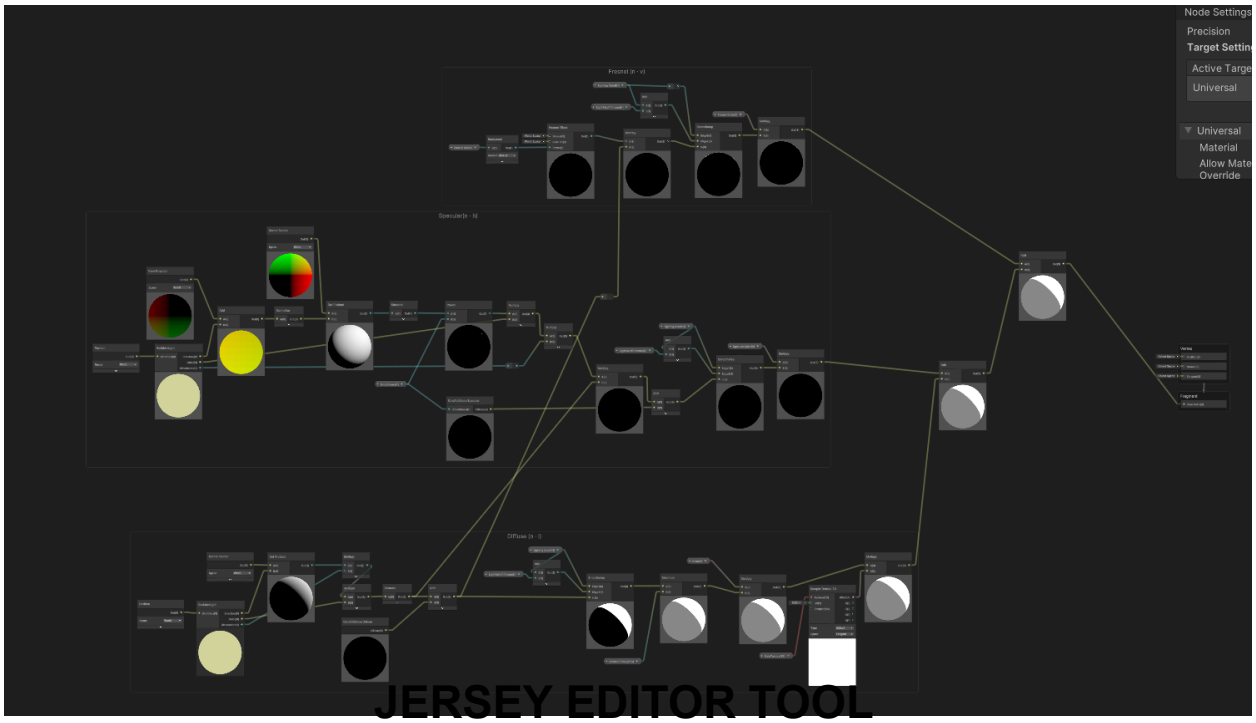


Fresnel (N · V): The fresnel section enhances object edges by calculating the angle between the surface normal and the view direction (N·V). This creates a view-dependent rim effect that becomes stronger toward the silhouette. The Fresnel Size parameter controls the thickness of the rim, while Smoothstep shapes how sharply it fades in. The result is multiplied by the Fresnel Color to provide artistic control over the rim lighting. This effect improves character readability, separates the model from the background, and strengthens the stylized aesthetic.



Combining all these nodes

Together, the diffuse lighting creates the primary cel bands, the specular component adds stylized highlight patches, and the fresnel term enhances silhouette edges. These components are blended to form a controllable, lightweight cel shading model that remains compatible with Unity's lighting system while achieving a clean stylized look.



JERSEY EDITOR TOOL

How Jersey Tool Works:

The Jersey Color Batch Tool is a custom Unity EditorWindow designed to streamline jersey color customization directly within the editor.

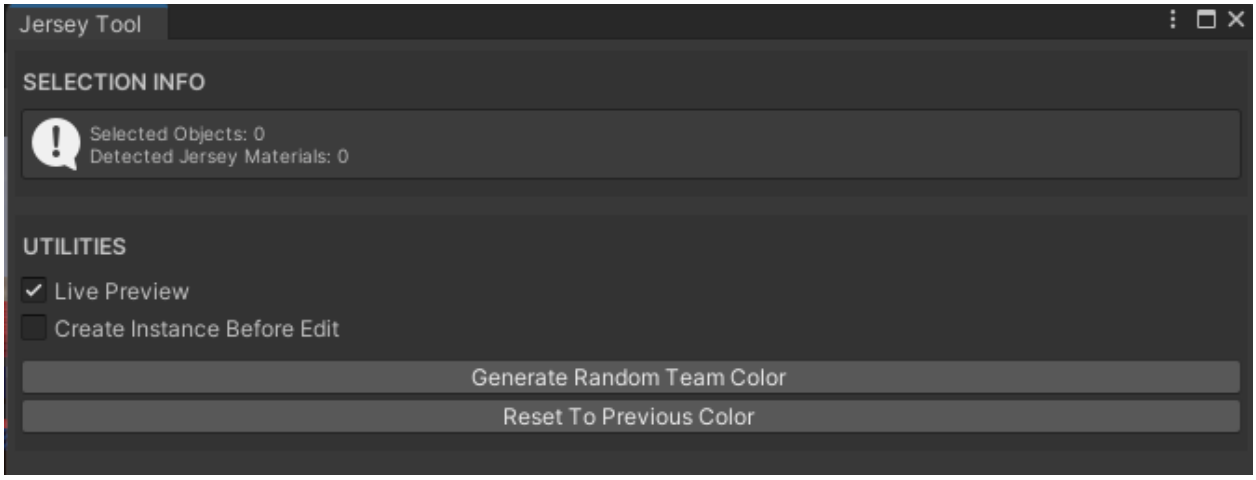
- It automatically detects selected objects and filters materials using a naming convention (**`Jersey_Cel_Mat`**).
- Color changes are applied through exposed shader properties with full Undo support.
- The tool supports batching, live preview, preset management, custom swatches, and team color history.

- Data persistence is handled using Unity's **`JsonUtility`**, where tool state (presets, swatches, history, toggles) is serialized into JSON and stored using **`EditorPrefs`**.
- On enable, the tool deserializes the saved JSON, restoring the previous session state seamlessly.

Tool Components & Features

Utilities:

- **Auto Detection of Jersey Material:** The tool scans selected GameObjects and their child Renderers, identifying materials whose names contain `Jersey_Cel_Mat`. This prevents unintended edits to other materials using the same shader.
- **Batching Editability:** All detected materials across multiple selected objects are edited simultaneously. This enables efficient team-wide color updates.
- **Live Preview:** When enabled, color changes from the picker are applied instantly without requiring manual confirmation.
- **Create Instance Before Edit:** Optionally duplicates shared materials before modification to prevent global material overrides.
- **Generate Random Team Color:**Generates HSV-based random colors within controlled saturation and brightness ranges for visually balanced results.
- **Reset Previous Color:** Restores the last applied color using stored `previousColor` data.

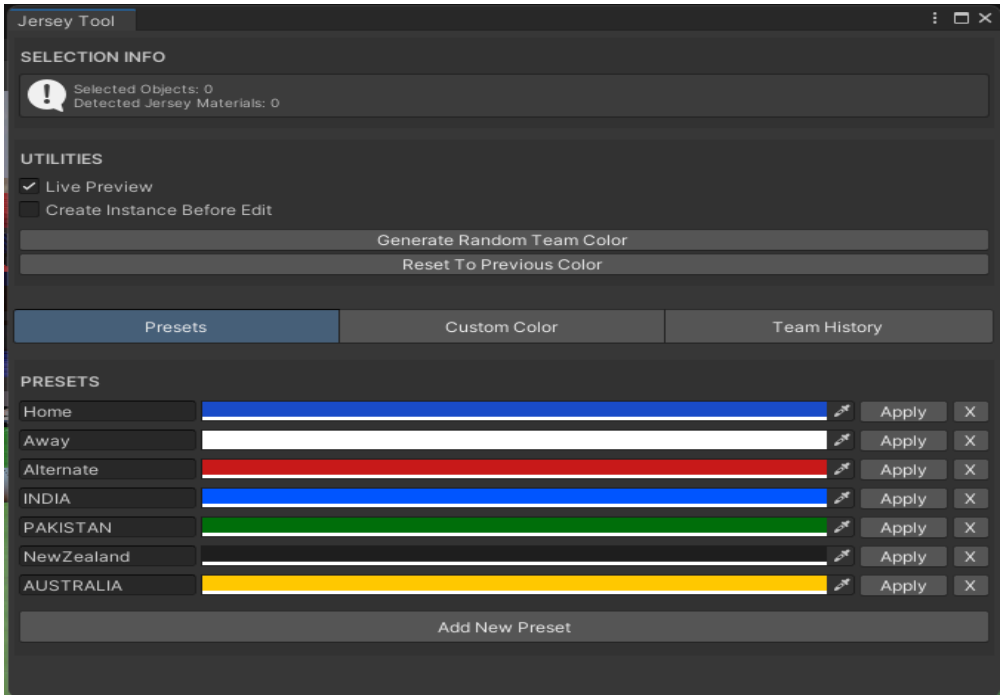


Presets

Presets allow predefined team colors with editable names and values.
Users can:

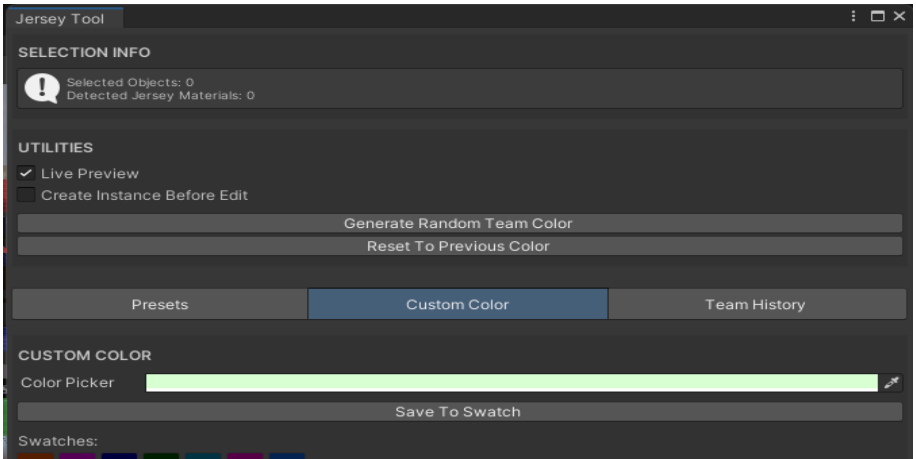
- Apply preset colors
- Add new presets
- Remove presets

Preset changes are saved automatically to JSON for persistence.



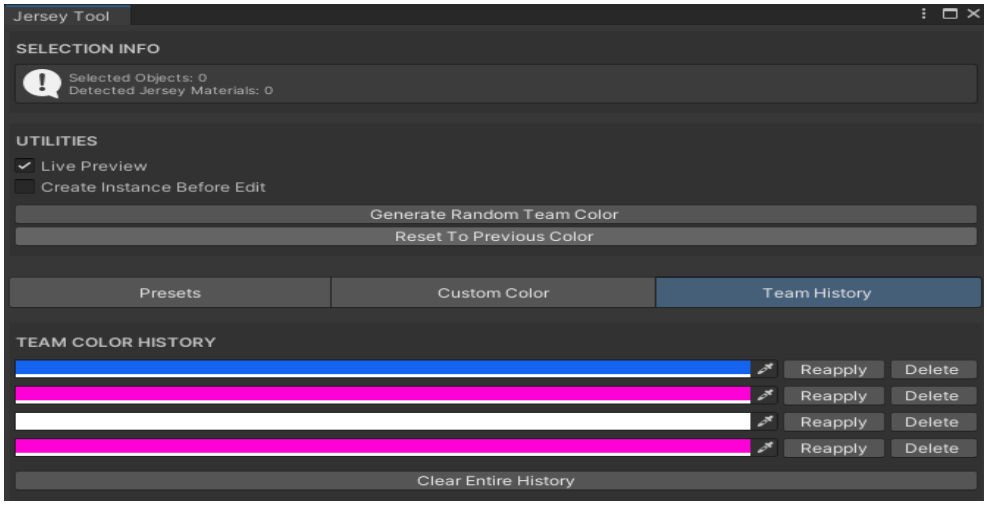
Custom Color:

- Color Picker: Allows manual color selection.
- Save to Swatch: Stores frequently used colors into a reusable swatch list.
- Swatches: Clickable color blocks that instantly apply saved colors and log them into team history.



Team History:

- Tracks recently applied team colors.
- Reapply:Quickly reapply a previously used color.
- Delete: Removes a specific color from history.
- Delete Entire History: Clears all stored history entries.



With all these features this tool becomes a production ready editor tool which saves time and increases reusability of the user.

THANK YOU FOR THE OPPORTUNITY!