VN may need some Effect and Transition shaders

A lightweight, versatile collection of shaders designed for **Visual Novels**, **2D games**, and **UI-based effects**.
This package provides a wide range of post-processing effects, transition shaders, helper scripts, and a demo scene to help creators quickly add polished visual effects to their projects.

0.Notice

This asset provides a collection of VN-oriented shaders and helper scripts.
A **free evaluation** version is available on GitHub for those who wish to preview the shaders before purchasing.

The Asset Store version offers the same content in a convenient, ready-to-use package, and purchasing it helps support future updates and development.

If you have any questions or feedback, feel free to contact me — I'm happy to help improve the asset.

1. Package Contents

1.1 Scripts

ShaderHelper.cs

A utility script designed to simplify the workflow of:

- Loading shaders via path
- Creating temporary materials
- Assigning shader properties dynamically
- Playing simple material animations

Note: DOTween is required for the animation methods in this version.
A coroutine-based alternative may be added in future updates.

ShaderTest.cs

A demo controller showing how to load shaders, assign parameters, and display various effects.

1.2 Demo Scene & Assets

- **Demo.Unity**
  A sample scene demonstrating shader effects and transitions.

- **Sample Sprites**

2. Effect Shaders Included

Barrel

- Barrel

- BarrelHyper

Blur

- GaussianBlur

- LensBlur

- MotionBlur

- RadialBlur

Color & Light Effects

- Glow
- Overglow
- Sharpen
- Mono
- ScreenFlickering
- BrokenTV

Distortion & Motion

- RandRoll
- RippleMove
- Shake

Artistic & Stylization

- Kaleido
- SpiralBackground

Water

- Water
- WaterAdvanced
- Ocean
- Rain

Speed Lines

- AnimeSpeedLine
- SpeedLine1
- SpeedLine2

3. Transition Shaders Included

- BlinkEye
- Circle
- CurtainPull

Fade

- Fade
- FadeSlideGlobal
- FadeSlideWave

Grid

- GridComplex
- GridTime

LerpSlices

- LerpSlices
- ClockWipe
- PolygonShrink
- Sawtooth
- Split
- Wave

4. Camera Shader

- **ZoomAndPan**
  A UI-friendly camera shader supporting zooming and panning without modifying your actual Camera component

5. Still Testing / Experimental (Optional Use)

These shaders are included but considered experimental:

- Color
- Colorless
- FadeSlideBlend
- Glitch
- GrayWave
- MaskedMosaic
- MixAdd
- Outline
- Overlay
- Ripple
- SDF
- SpiralWipe
- SquareTransition
- VenetianBlinds

They may be refined or removed in future updates.

6. How to Use

You may reference `ShaderHelper.cs` for details.
Basic usage involves:

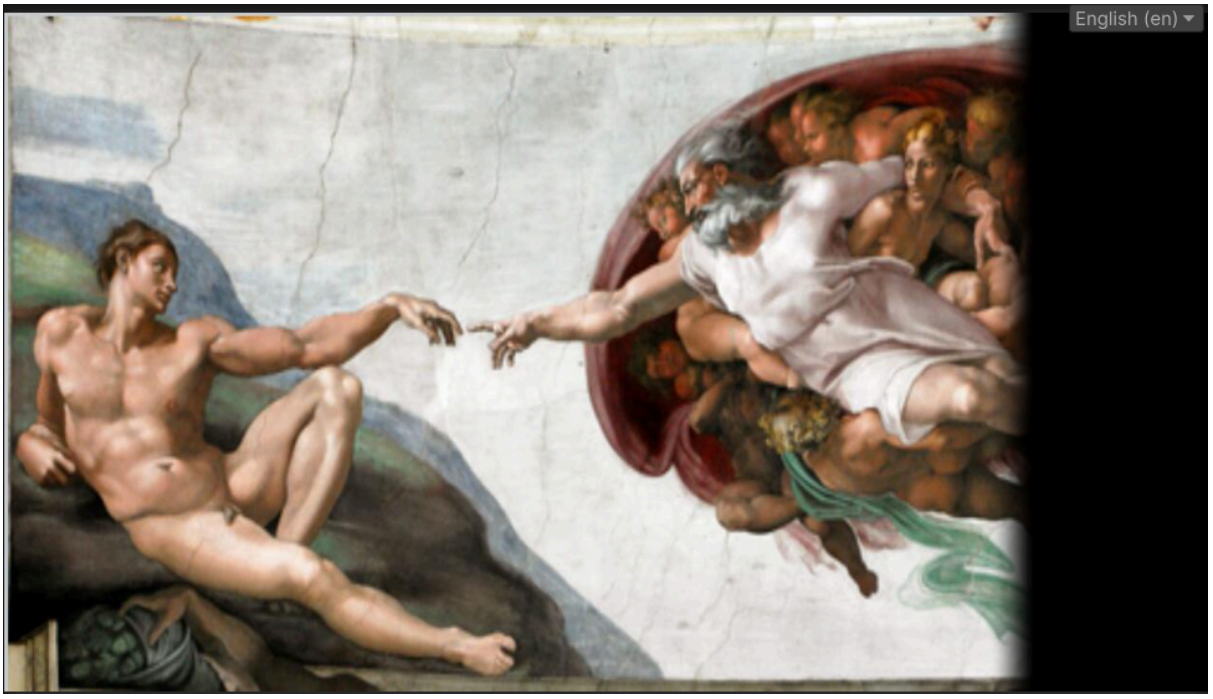
1. Loading a shader via path
2. Creating a material
3. Passing parameters using `Dictionary<string, object>`
4. Applying the effect to an Image or Renderer
5. Running animations or transitions if desired

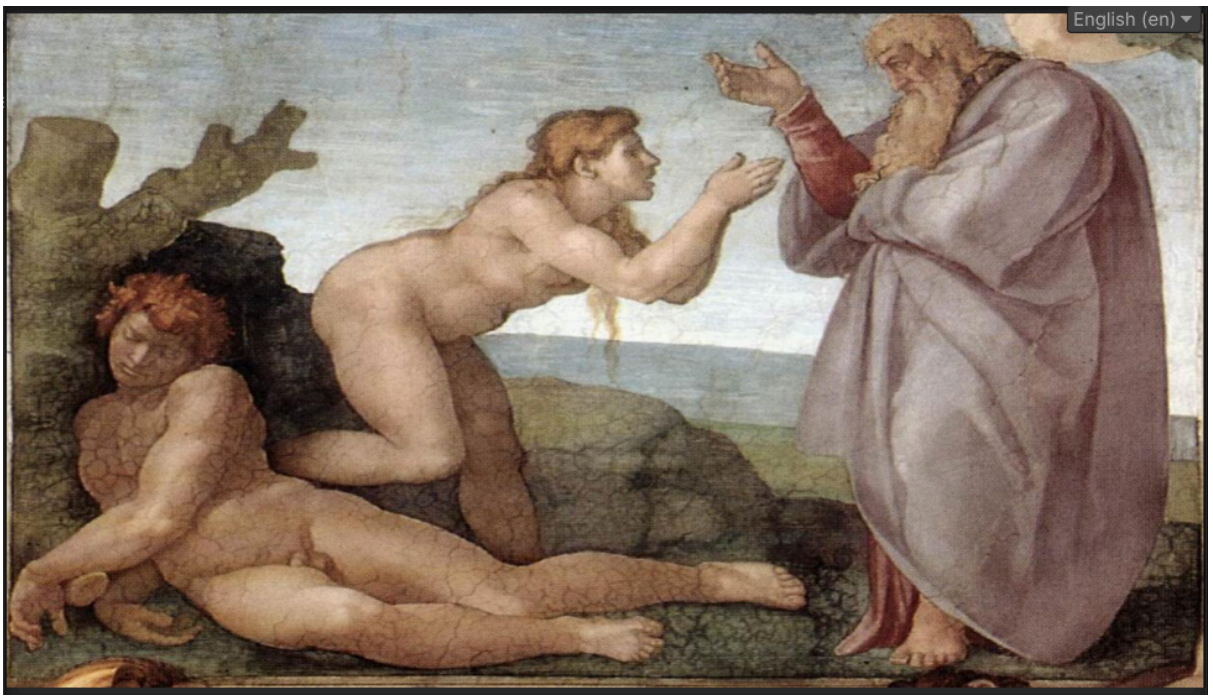
More examples can be found in the demo scene and scripts.

> **Note:**
> If you wish to load shaders through Resource paths, ensure the shader folder is placed under:
> `Assets/Resources/`
>
> Alternatively, modify the loading method in `ShaderHelper.cs` to match your project's file structure.

```
Unity Message | 0 references
private void Update()
{

    if (Input.GetKeyDown(KeyCode.G)) ApplyFadeSlideWaveTransition();

}


1 reference
void ApplyFadeSlideWaveTransition()
{
    Sprite originalSprite = TargetImage.sprite;

    var request = new ShaderAnimationRequest(
        shaderName: TransitionShader.FadeSlideWavePath, // Shader Path
        targetImage: TargetImage,// Shader Path
        targetSprite: TargetSprite // The Sprite we want to set
                                   // after the transition is here set to "CreationOfEva".

     //  duration: AnimationDuration // Total transition time, you can custom it
    );

    ShaderHelper.TransitionAnimation(request); // Apply transition animations
}
```

The following Transition will be:

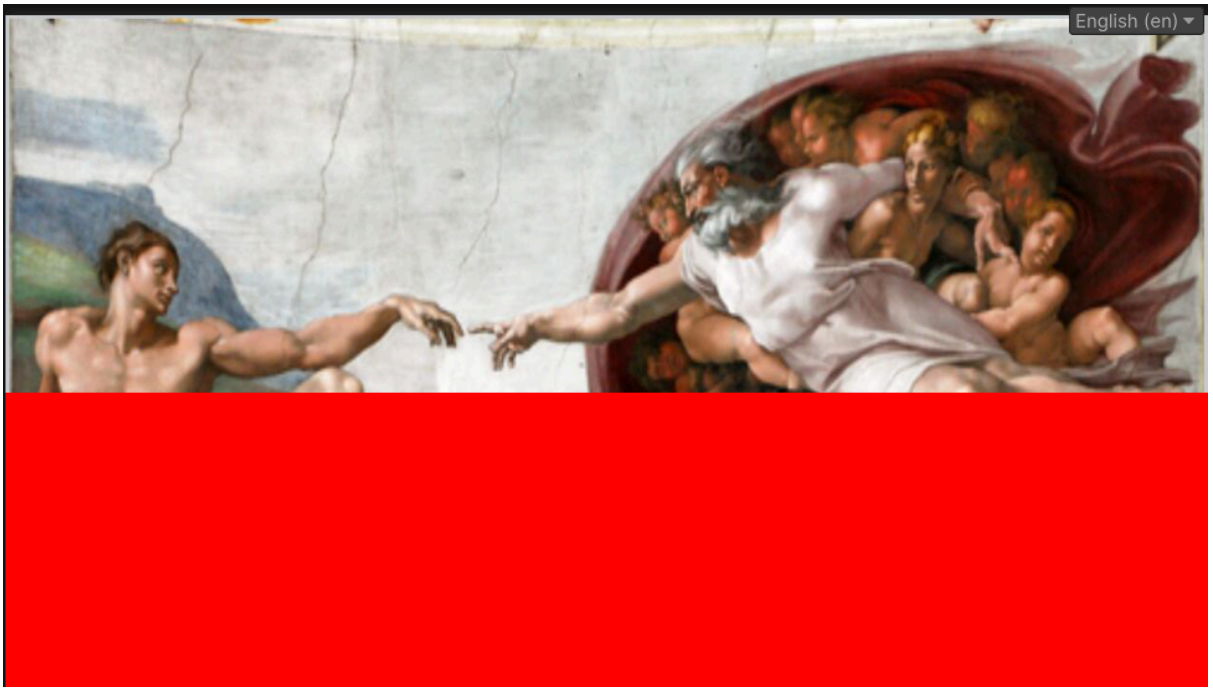


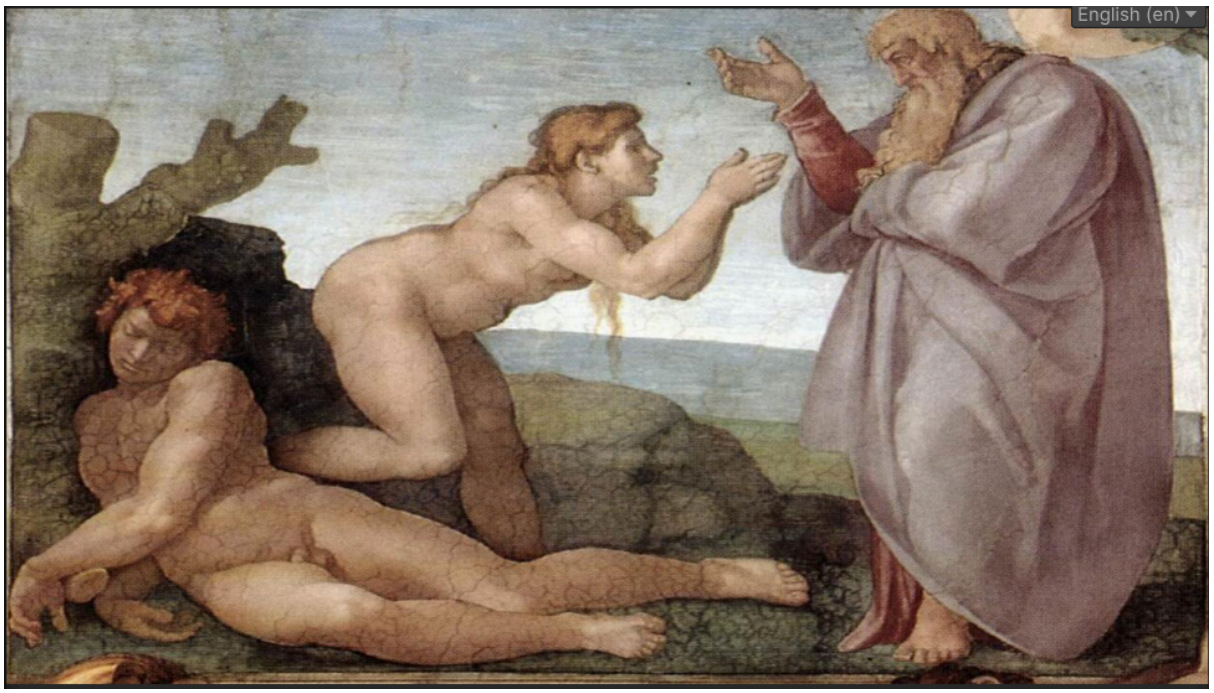


Or, we can achieve different effects by making the following changes to the code without creating additional materials in the project.

```
Unity Message | 0 references
private void Update()
{

    if (Input.GetKeyDown(KeyCode.G)) ApplyFadeSlideWaveTransition();

}


1 reference
void ApplyFadeSlideWaveTransition()
{
    Sprite originalSprite = TargetImage.sprite;

    var request = new ShaderAnimationRequest(
        shaderName: TransitionShader.FadeSlideWavePath, // Shader Path
        targetImage: TargetImage,// Shader Path
        targetSprite: TargetSprite, // The Sprite we want to set
                                    // after the transition is here set to "CreationOfEva".
        customParams: new Dictionary<string, object>
        {
            { ShaderProperty.TransitionColor, Color.red },              // The color that appears during the transition
            { ShaderProperty.Pivot, new Vector4(0.5f, 0f, 0, 0) },        // Transition starting point (Form bottom)
            { ShaderProperty.Direction, ShaderPropertyValue.Vertical },  // Slide direction: vertical
            { ShaderProperty.SoftEdge, 0f },                             // Hard edge (no soft fade)
        },

        duration: 10f // Total transition time, you can custom it
    );

    ShaderHelper.TransitionAnimation(request); // Apply transition animations
}
```

It will be like this:





7. Credits & Acknowledgments

Some shaders in this package are **inspired by publicly available shader tutorials and reference materials**.
All included shaders have been **rewritten, adapted, or optimized** for easier use in Visual Novel and 2D UI environments.

If any original author wishes their referenced concept to be removed, please contact me and I will update the asset immediately.

Thank you for your support — wishing you success in your development!