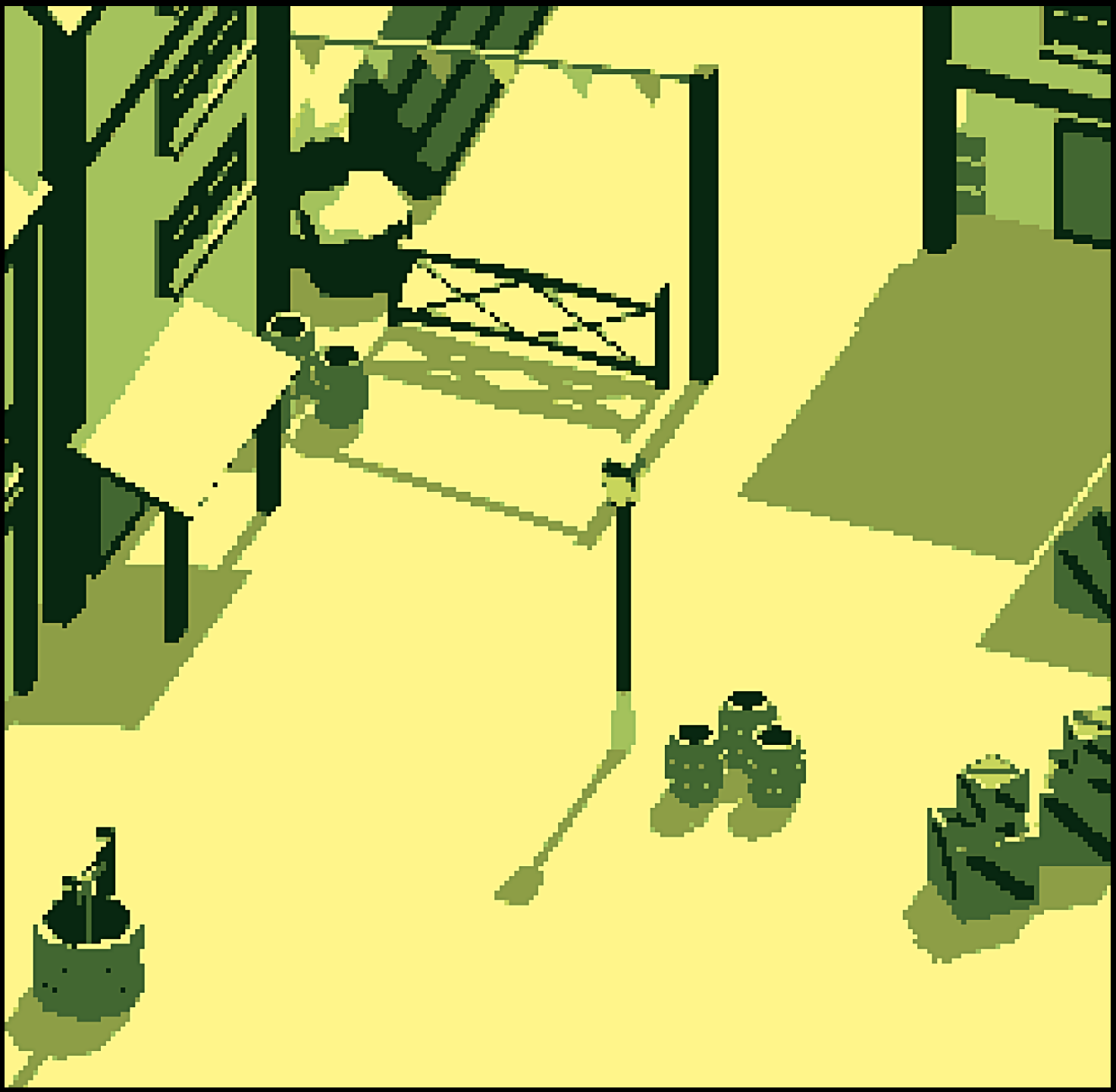
DOCUMENTATION

PIXEL STYLIZER



Content

1 Getting Started

1.1 How does it work?

This asset uses 1 script and 2 simple shaders to redefine the color palette of your camera. The first shader pixelates your camera view without touching the colors of it. The second shader uses 9 colors (defined by the user) to set a new color palette for the camera. It works for either 2D or 3D projects.

Your objects on screen (characters, buildings, vegetation, etc.) will be repainted with 8 of these 9 colors. The last color will repaint the shadow/darkest part of the object.

How exactly do I know how this script is painting each part of my game?

I am going to explain it by using an example. Let's suppose that you have a cube on your scene. Without this script, your cube displays a white color. Then, you add "Pixel Stylizer Camera" to your project. You set up a 3 color palette: Pastel yellow, Red, and Dark Blue. By doing this, you suddenly see that your cube displays a pastel yellow color. Well, what happened here is that the script tried to find the nearest (more alike) color from the chosen palette to the real color of the cube, and then it displays that color. That is the way in which every object in your scene is repainted.

It is important to notice that this script do not change any texture or material value. It just adjust the camera view to a new color palette.

Before moving onto the tutorial, I would like to let you know that every script and shader of this asset is commented so you can clearly see what is every part doing, and if you want, you can change the default values from inside the script.

2 Tutorial (Importing to your project)

In this part you will learn step by step how to implement this asset in your scene.

1. **Localize your main camera**. Select your main camera, then, from the inspector bar click on "Add Component". After this, write "Pixel Stylizer Camera" and select it. It is important to make sure your camera is tagged as "MainCamera", otherwise, you will get an error.

   You will notice a palette color change by default.

2. **Screen Settings**. Once you have added this script to your camera, you will see a label which text is "SCREEN". In this part, you will define the screen size reference for your project. You can choose from 320 px up to 4096 px wide. This part is completely related to the next variable.

   The next variable is "Pixel Size". You can choose a size from 1px to 16 px. You will notice a bigger pixelated looking if your screen reference size is small.

   If you select your pixel size as 1px, you will see an extra option called "Smooth

camera". If you set it to "true", your camera will have a bilinear filter, if it is set "false", your camera will have a point filter (better for pixelated games). This option was thought for people who do not want to pixelate their scene but use only the stylizer.

3. **Camera Settings**. This part is divided into 2 major groups: "Presets" and "Custom".

   In the "presets" tab you will be able to choose from 1 of the 11 included palette presets to stylize your game. **Perhaps, at first sight your scene will not look as you want.** I recommend you to make small changes to the color values of the chosen preset so it adapts well to your game scene. However, it is important to know that if you change from one preset to another, all the changes you made will be lost.

   In the "custom editor", you can create a color palette from 2, 3, 4 or 8 different colors. If you make changes you can always recover the previous values with "Ctrl + Z" as long as you stay in the same scene.

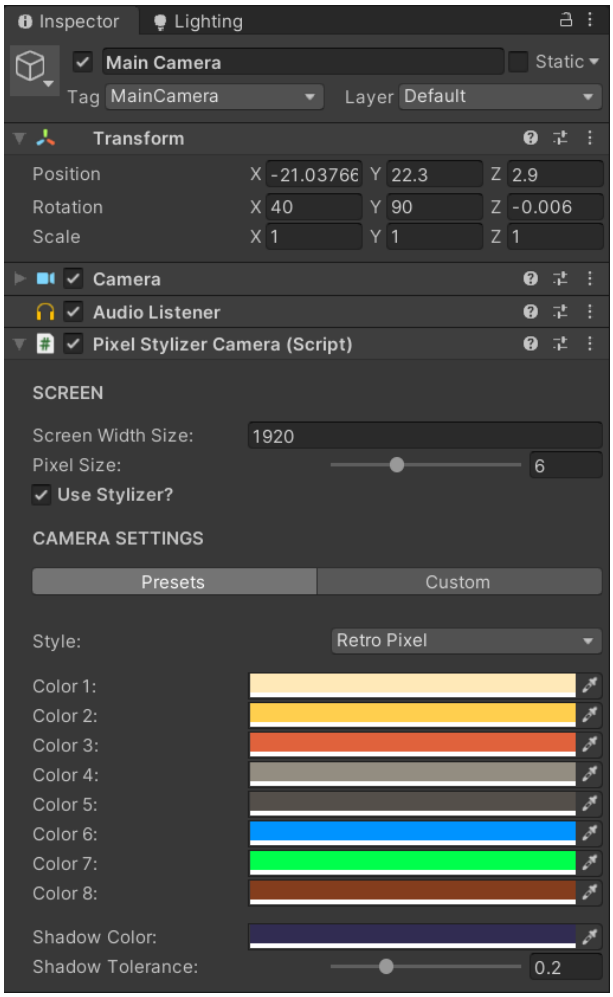


Figure 1: *Editor window of "Pixel Stylizer Camera."*

3 Scripting

3.1 Public Variables.

3.1.1 Camera Variables.

screenWidthSize (int)

This variable is used to define the screen width reference of the camera view. It can take values from 320px to 4096px.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.screenWidthSize = 1280;
    }

}
```

pixelSize (int)

This variable is used to define the pixel size of your game. It can take values from 1px to 16px.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.pixelSize = 4;
    }

}
```

smoothCamera (bool)

If this is set "true", the camera will have a bilinear filter, if it is set "false", the camera will have a point filter. The effects of this variable are only seen if your pixelSize is 1.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      //The effects of this variable are only seen if your pixelSize is 1.
      PSC.smoothCamera = true;
    }

}
```

stylizePixels (bool)

If this is set "true", the camera will display a color palette from a preset or a custom palette, if it is set "false", the camera will only pixelate the game view without changing any color.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.stylizePixels = true;
    }

}
```

3.1.2 Color Variables.

cameraType (int)

If this is equal to 0, the camera will display a color palette from a preset, if it is equal to 1, the camera will display a custom color palette. It is important to give only these 2 values: 0 or 1.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      //The camera will display the custom palette.
      PSC.cameraType = 1;
    }

}
```

presetStyle (int)

This variable defines what preset is active. It takes values from 0 to 10, where the 0 displays the first preset of the inspector list called "Retro Pixel" and the 10 displays the last preset of the inspector called "Army".

This variable should be used with the public method "SetPreset(int presetIndex)" to update the preset on scene.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      //The camera will display the custom palette.
```

```
      PSC.presetStyle = 1;
      PSC.SetPreset(PSC.presetStyle);
    }

}
```

Color_1 (Color) → Color_8 (Color)

These variables represent the 8 colors used by the presets. You can change these values from script, however, after changing between scenes, the colors will go back to their initial value.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.Color_1 = new Color32(255, 233, 184, 255);
      PSC.Color_2 = new Color32(255, 207, 78, 255);
      PSC.Color_3 = new Color32(224, 98, 60, 255);
      PSC.Color_4 = new Color32(147, 141, 129, 255);
      PSC.Color_5 = new Color32(84, 79, 74, 255);
      PSC.Color_6 = new Color32(0, 147, 255, 255);
      PSC.Color_7 = new Color32(0, 255, 76, 255);
      PSC.Color_8 = new Color32(132, 61, 29, 255);
    }

}
```

shadowColor (Color)

This is the color variable used by the presets. You can change its value from script, however, after changing between scenes, the color will go back to their initial value.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

```

```
    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.shadowColor = new Color32(49, 44, 82, 255);
    }

}
```

customColor_1 (Color)

These variables represent the 2, 3, 4 and 8 colors used by a custom preset created by the user. You not have to run any special method to see the color changes on screen. Just edit these variables from inspector or code.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.customColor_1 = new Color32(240, 240, 240, 255);
      PSC.customColor_2 = new Color32(210, 210, 210, 255);
      PSC.customColor_3 = new Color32(180, 180, 180, 255);
      PSC.customColor_4 = new Color32(150, 150, 150, 255);
      PSC.customColor_5 = new Color32(120, 120, 120, 255);
      PSC.customColor_6 = new Color32(90, 90, 90, 255);
      PSC.customColor_7 = new Color32(60, 60, 60, 255);
      PSC.customColor_8 = new Color32(30, 30, 30, 255);
    }

}
```

customShadowColor (Color)

This is the color variable used by the custom palette editor.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
4
5  public class MyScript : MonoBehaviour{
6
7      public PixelStylizerCamera PSC;
8
9      // Start is called before the first frame update
10     void Start()
11     {
12       PSC.customShadowColor = new Color32(49, 44, 82, 255);
13     }
14
15 }
```

shadowTolerance (float)

This variable is used to indicate how dark must be an object in order to repaint it with the shadow color. It should take values from 0.05f (a very dark object) to 0.6f (a little bright object).

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class MyScript : MonoBehaviour{
6
7      public PixelStylizerCamera PSC;
8
9      // Start is called before the first frame update
10     void Start()
11     {
12       PSC.shadowTolerance = 0.3f;
13     }
14
15 }
```

3.2 Public Methods.

SetPreset (int presetIndex)

If you are using a stylizer preset, you can change to another preset with this method. It takes one argument: presetIndex, which is the number of preset you want to select. Its value go from 0 to 10, where 0 is "Retro Pixel" and 10 is "Army".

```
1  using System.Collections;
2  using System.Collections.Generic;
```

```
using UnityEngine;

public class MyScript : MonoBehaviour{

    public PixelStylizerCamera PSC;

    // Start is called before the first frame update
    void Start()
    {
      PSC.presetStyle = 3;
      PSC.SetPreset(PSC.presetStyle);
    }

}
```

4 Warnings

In this part I would like to tell you some things you should consider when using this asset:

1. **Editing Presets**. You can edit the colors of a preset from inspector in order to get a nice looking result on your project. I scripted that way because every game is different and it is not possible to define a color palette that suits for every game. However, if you change to another preset, "Ctrl + Z" will not recover the changes made to that preset. You could change the preset value from script to avoid this.

2. **Post Processing/URP/LWRP**. This asset will not work if you are either working with post processing effects or URP/LWRP environments.