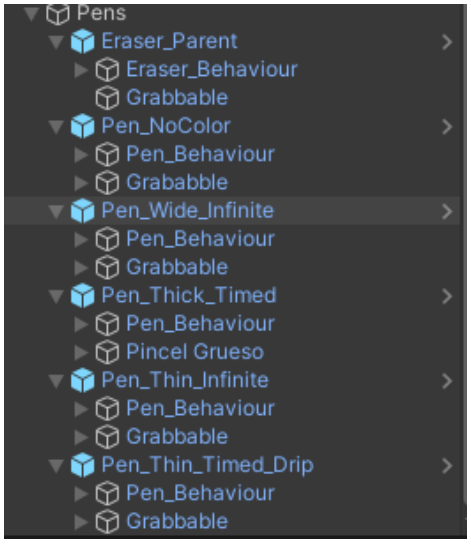
SimpleInk VR

Starting

With SimpleInk VR you can easily implement in your Virtual Reality Game or Experience an entertaining and immersive way for the users to paint or take note with just a few steps. Mainly focusing on ink based Paint styles (Dripping Ink that can stain the paper, having to recharge your pencil when you have painted for a certain type, using different colors to paint) with all features openly available for the developer to design your own painting spot. Within this guide we will see a standard setup for this asset and the procedure to create your own surface to paint on; in only a few steps.

Standard Setup

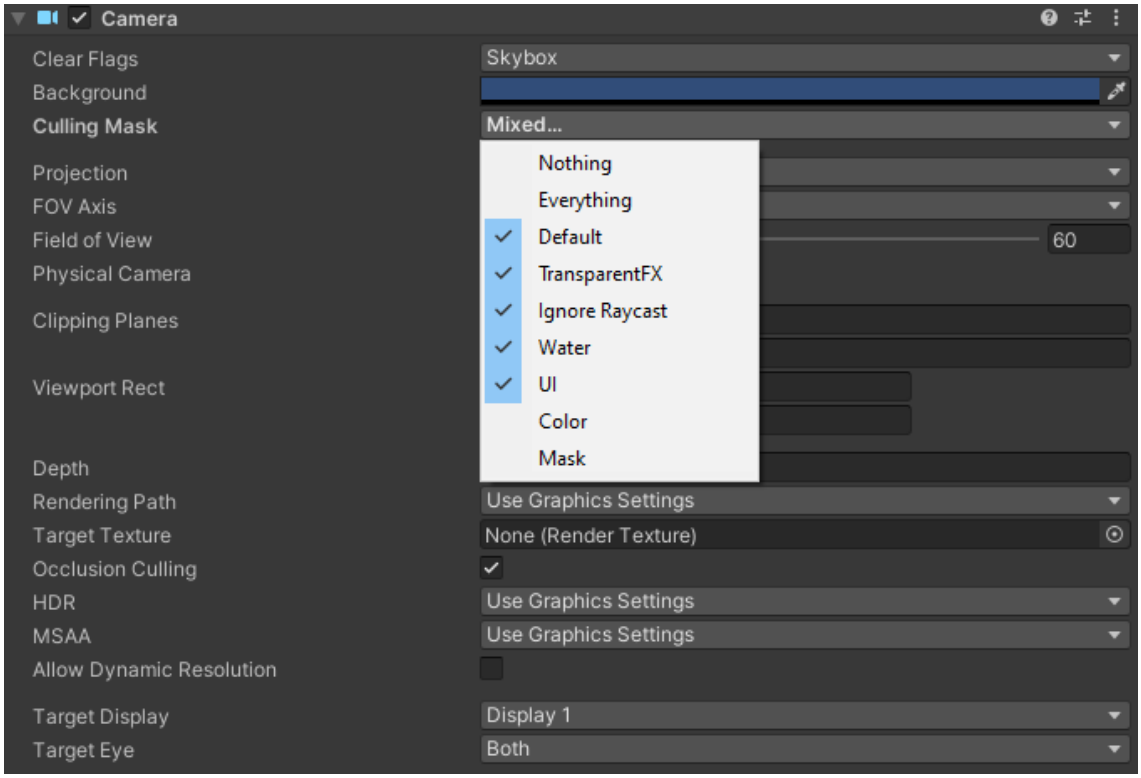
The easiest way to implement this Assets is with SimpleInkVR>Prefabs>Paint_Set as it brings all the prefabs you need to get yourself started. It has two different pages, with different materials and different aspect ratios (Squared, 1:1; and rectangular, 16:9). Also has 5 different configurations for the pencils, one eraser and Inks from each one of the primary and secondary colors, besides white and black. Alongside with these, there is a mixer for the players to create their own color, and a saver to store the colors they like. The only thing needed for this prefab to work is to include the Grabbable Script (Depends on what SDK you are using; i.e. OculusSDK has OVRGrabbable, XR Toolkit from Unity uses XR Grab Interactable… etc.) into the "Grabbable" Object on each pen. In case you scale the paper object, you need to resize the cameras. (More info in Papers)



In order to keep this asset as polyvalent as we could, we decided to no include the Grabbable Script as it delegates in libraries from SDK and might give some error while exporting. You just need to treat the "Grabbable" as any other Grabbable object in your scene. The Pen_Behaviour follows the children of Grabbable, "BallEnd1", and controls all the painting feature. Lastly, you have to include the following layers, and make sure the cameras are not rendering those layers:

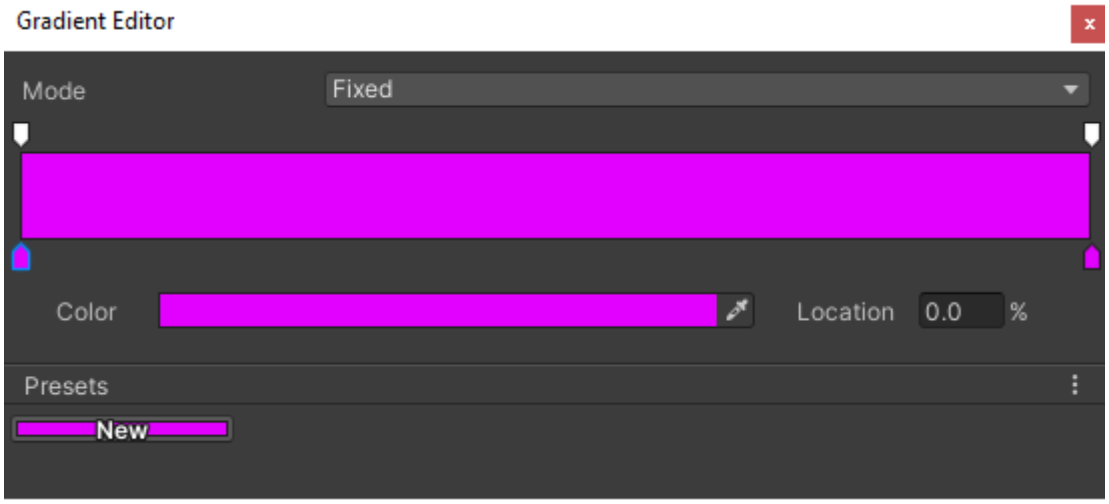
30: Color

31: Mask


Camera
Clear Flags
Skybox
Background
Culling Mask
Mixed...
Nothing
Everything
Default
TransparentFX
Ignore Raycast
Water
UI
Color
Mask
Projection
FOV Axis
Field of View
60
Physical Camera
Clipping Planes
Viewport Rect
Depth
Rendering Path
Use Graphics Settings
Target Texture
None (Render Texture)
Occlusion Culling
HDR
Use Graphics Settings
MSAA
Use Graphics Settings
Allow Dynamic Resolution
Target Display
Display 1
Target Eye
Both

Ink Modifications

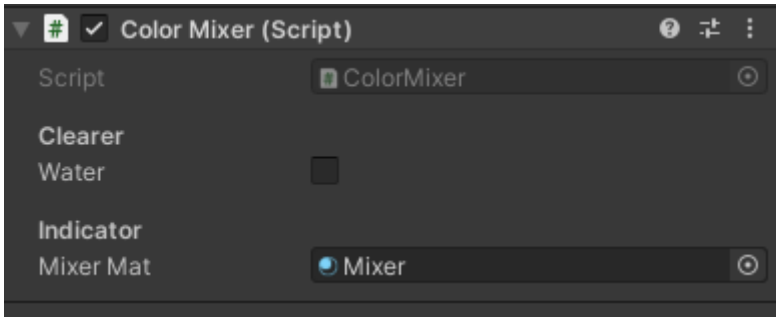
For the colors that are already created, the settings are pretty simple. You just have to set the whole gradient with a single color and when the pencil enters (if there is color activated in the pencil) it will now paint in that color. This script is on the “Ink” children on InkParent for each object.



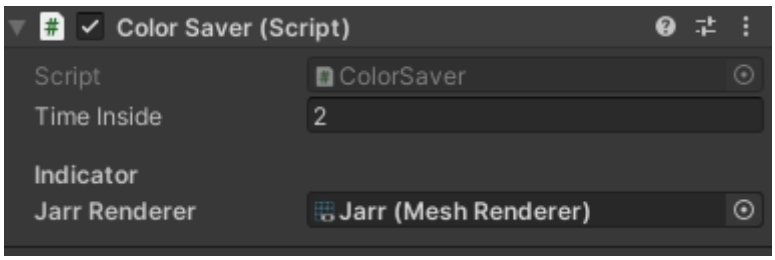
You can also use whole gradients with different colors for a more experimental effect.

Savers and Mixers

You do not have to give all colors to the player, in case you want to have them create their own colors, we included a mixer and a saver for this feature. The mixer ink has 2 different uses; you have one Boolean that changes from mixer to water. When a pencil enters in water it clears all the color inside of it and resets the color to its starting point. When the Boolean is no active, the mixer will blend both color and set the result into the pencil, and also the material of the jar, so the player can know the result of the mix.

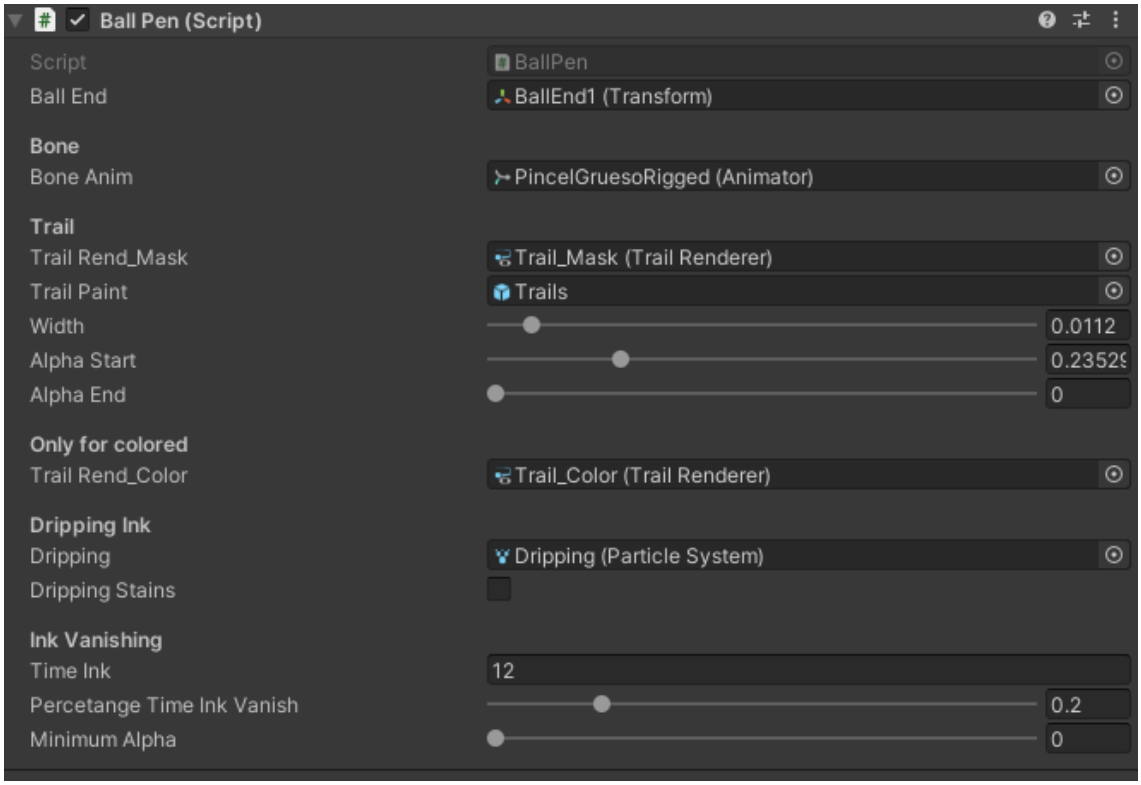


If the player is satisfied with the color it can be stored into the Savers. These work in a very similar way, if the pencil stays inside the ink for a time longer than the TimeInside, the color saved will be replaced for the actual color in the pencil, and that same color will be applied to the mesh renderer of the jar. If the pencil exits before that time, it will take the color saved in the Color Saver.



Pencil features

It is within the pen behaviour where we find the widest choices to personalize our painting feature, so let us take a close look at all the parameters you can edit in this script.



BallEnd is the reference to the tip of the pencil. This allows the colliders to stay always around that object for a more realistic feature.

The bone animator for the movement of the pencil. When moving the pencil in a paper surface the pencil will move its tip in the opposite direction of the movement, as in real life. In case your model does not have a bone animation, you can just leave null this reference.

Trail Rend_Mask is a reference for the rail renderer that paints the render texture, and TrailPaints is the parent for both Rend_Mask and Rend_Color.

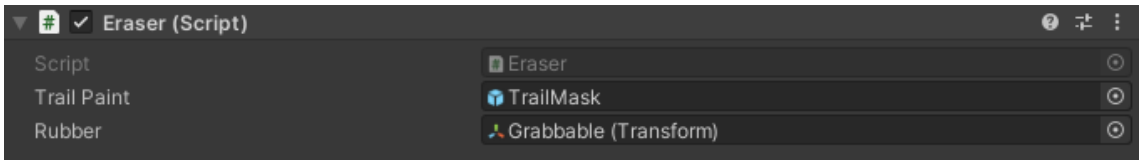
Width, AlphaStart and AlphaEnd are variable of the TrailRenderer_Mask, so modifying the color gradient or width of the TrailRenderer will be overwritten by this script. AlphaStart and AlphaEnd are used for a more diffuse painting from the pencil.

When referencing a Trail_Color the pencil will paint not only black and white but also in colors. We recommend to always use color and force it to black rather than not having colors, but in case you do not want any color you can leave this reference null.

Dripping Ink is for an effect of the dripping when you wet your pencil in Ink. If you do not want this effect, you can leave the reference to null. Dripping Stains will make the particles to also paint the paper when activated. This can only be modified before the start of scene.

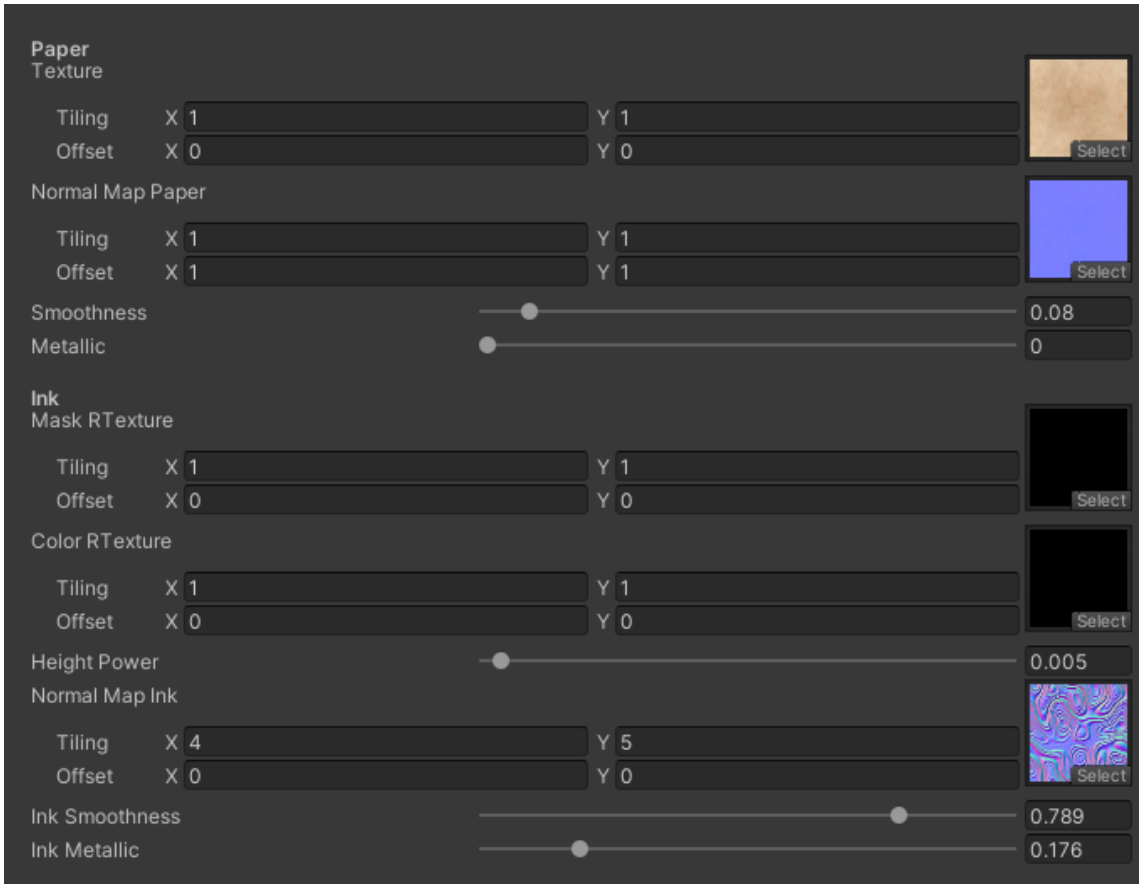
Ink vanishing is a feature to limit the time you can paint without refilling the ink. You can leave the Time Ink value to 0 for infinite time. If left with any time, when the Percentage of Time Ink Vanish happens, the pencil will start decreasing its alpha until It reaches the Minimum Alpha value.

For erasers the setting is much easier. You also need to set the Grabbable Script into the Grabbable Object of the Eraser. The references of this script are just the Trail Renderer that will paint over the previously painted surfaces and the Grabbable object to follow. Adjusting the alpha in this TrailRenderer could apply a more realistic effect.



Papers

Paper are based on mainly two things, a shader with (or without) normal, and two render textures, one for color and other one as a mask of what is painted or not.



The first texture and normal is for the paper with no paint on it. Same applies for the smoothness and metallic in the first section. The second section is for the render textures (if different materials have the same render texture when you paint in one of them, it will be painted in both; This can be used as a feature or solve it easily by duplicating render textures). Height power is the difference of heights between painted and not painted paper. The second normal map will only be applying to the painted zones. This creates a thicker effect on painted spots. Lastly we have smoothness and metallic values for the painted spots, for a more realistic result.

To create your own paper with the same ratio that is already created you have to do the following:

1. Duplicate Render Texture folder of the Ratio you want to use
2. Change the name for Mask and Color Render texture
3. Create a Material that uses both render textures
4. Create a plane with the material and add as it children two cameras
5. Both cameras have to render only the proper layer, 31: Mask for the mask render texture and 30: Color for the color render texture. Both of them have to be orthographic size, Clear flags set to “Solid Color” and they need to have a CameraFlagsChange script.
6. Adjust the size of the camera to fit the size of the plane

For a custom ratio, you need to do a few steps before the beforehand mentioned:

- Create a plane with the same ratio as the Render texture you want to create.
- Create two render textures with unique names and set the width and height to keep the desired ratio.
- Follow from Step 3 of the previous guide.