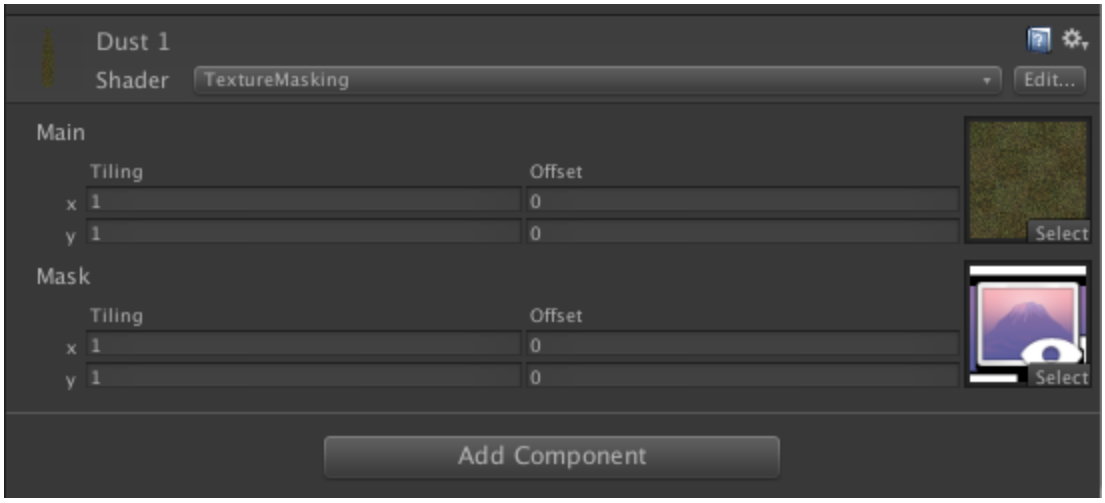
How to remove dust/fog from an object in Unity with swipe?

Objective -:
The Main objective of this Code sample is to create a dust/fog removing effect using Unity. Using, Texture Masking shader, Mask Construction Shader, Render Texture, Camera Masking Script and masking camera, you can create your own effect. Read on to know how and download the source code. It's free!!

Texture Masking Shader -:

This shader gives an effect of a layer of dust on the screen. We only need to add dust texture and render texture (as Mask)in material shader.



Shader-:

```
Shader "TextureMasking" {
   Properties {
      _MainTex ("Main", 2D) = "white" {}
      _MaskTex ("Mask", 2D) = "white" {}
   }

   SubShader {
      Tags {"Queue"="Transparent" "IgnoreProjector"="True"
"RenderType"="Transparent"}
      ZWrite Off
      ZTest Off
      Blend SrcAlpha OneMinusSrcAlpha
      Pass {
         CGPROGRAM
         #pragma vertex vert
         #pragma fragment frag
         #pragma fragmentoption ARB_precision_hint_fastest
         #include "UnityCG.cginc"

         uniform sampler2D _MainTex;
         uniform sampler2D _MaskTex;
         uniform float4 _MainTex_ST;
         uniform float4 _MaskTex_ST;

         struct app2vert
         {
            float4 position: POSITION;
            float2 texcoord: TEXCOORD0;
         };

         struct vert2frag
         {
            float4 position: POSITION;
            float2 texcoord: TEXCOORD0;
         };
```

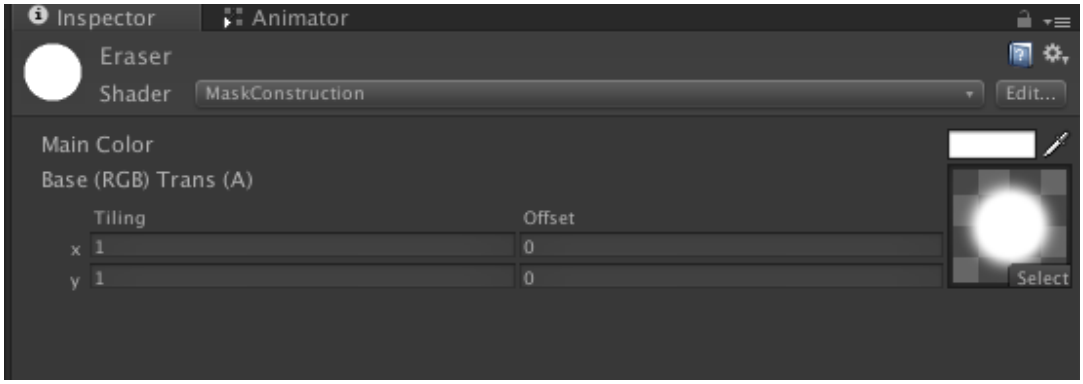
```
        vert2frag vert(app2vert input)
        {
            vert2frag output;
            output.position = mul(UNITY_MATRIX_MVP, input.position);
            output.texcoord = TRANSFORM_TEX(input.texcoord, _MainTex);
            return output;
        }

        fixed4 frag(vert2frag input) : COLOR
        {
            fixed4 main_color = tex2D(_MainTex, input.texcoord);
            fixed4 mask_color = tex2D(_MaskTex, input.texcoord);
            return fixed4(main_color.r, main_color.g, main_color.b, main_color.a * (1.0f -
mask_color.a));
        }
        ENDCG
    }
  }
}
```

How to use this shader?

- Start by creating a sprite in Hierarchy panel .
- Add dust image (as Sprite) in this Sprite
- Add new Material in this sprite and select Texture Masking Shader.
- In this material, set Dust Texture as Main Texture and Mask as Render Texture.
- For Render Texture, create a Render texture in project Panel and Select this render texture in this material.

Mask Construction Shader -:

This shader gives a wiping off dust effect each time the user swipes the screen. To make it happen, we also need to add an eraser image and set it's colour too.



Shader-:

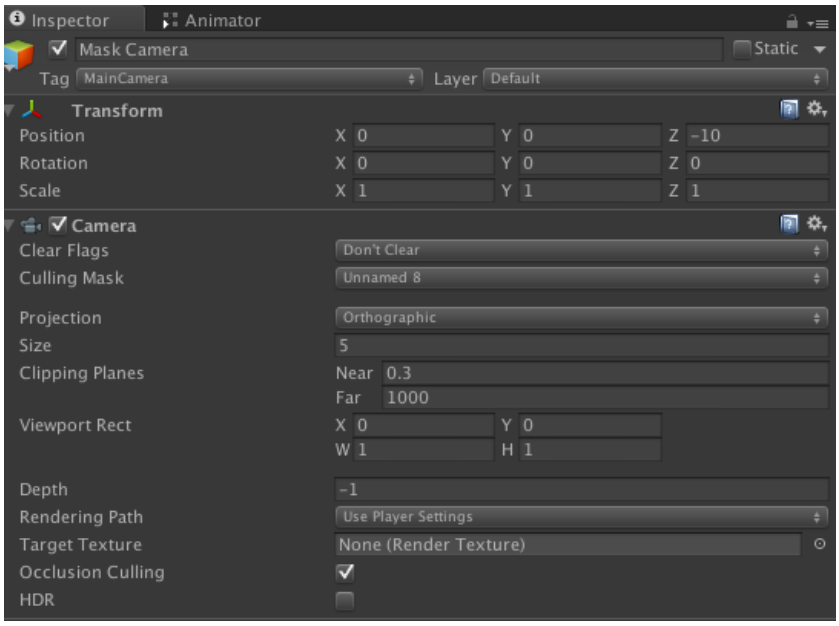
```
Shader "MaskConstruction" {
   Properties {
      _Color ("Main Color", Color) = (1,1,1,1)
      _MainTex ("Base (RGB) Trans (A)", 2D) = "white" {}
   }

   SubShader {
      Tags {"Queue"="Transparent" "IgnoreProjector"="True"
"RenderType"="Transparent"}
      ZWrite Off
      ZTest Off
      Blend One One
      BlendOp Max
      Pass {
         Lighting Off
         SetTexture[_MainTex] {
            matrix [_UVMatrix]
            ConstantColor [_Color]
            combine texture * constant
         }
      }
   }
}
```

How to use this shader?

- In Project Panel Create an empty material
- Select Shader as “MaskConstruction”
- Select Main Color and Select Erase image.

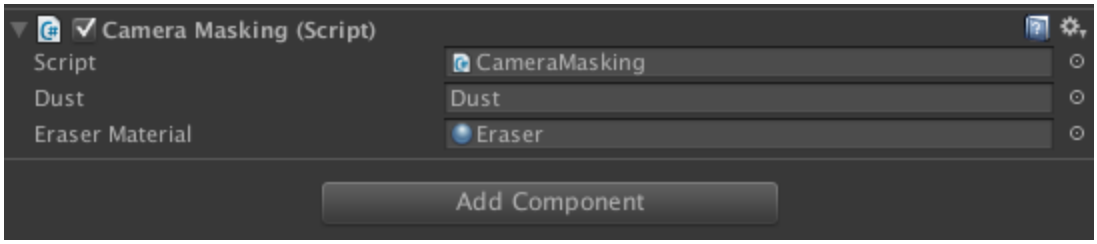
Masking camera-:

Masking Camera shows dust on screen as Mask.



How to Use Masking Camera ?

- In Hierarchy panel create a camera.
  - In this Camera component as Clear Flag Select ”Don’t clear”
  - Create a layer and name it as “Mask ” and set camera Culling mask select as “Mask”.
  - Set Projection to “Orthographic”.
  - and its size “5”
  - add “**Camera Masking** ” script on this camera object



- Drag dust object (Sprite) to get the dust effect on Screen.
- Drag Eraser Material to wipe off the aforementioned sprite and create the effect.

Camera Masking (Script)-:

(here come full script)

```
using UnityEngine;
using System.Collections;

public class CameraMasking : MonoBehaviour
{
    public GameObject Dust;
    Rect ScreenRect;
    RenderTexture rt;
     Texture2D tex;
    public Material EraserMaterial;
    private bool firstFrame;
    private Vector2? newHolePosition;

    private void EraseBrush(Vector2 imageSize, Vector2 imageLocalPosition)
    {

        Rect textureRect = new Rect(0.0f, 0.0f, 1.0f, 1.0f); //this will get erase material texture
part
        Rect positionRect = new Rect(
            (imageLocalPosition.x - 0.5f * EraserMaterial.mainTexture.width) / imageSize.x,
            (imageLocalPosition.y - 0.5f * EraserMaterial.mainTexture.height) / imageSize.y,
            EraserMaterial.mainTexture.width / imageSize.x,
            EraserMaterial.mainTexture.height / imageSize.y
        ); //This will Generate position of eraser according to mouse position and size of eraser
texture
```

```
        //Draw Graphics Quad using GL library to render in target render texture of camera to
generate effect
        GL.PushMatrix();
        GL.LoadOrtho();
        for (int i = 0; i < EraserMaterial.passCount; i++)
        {
            EraserMaterial.SetPass(i);
            GL.Begin(GL.QUADS);
            GL.Color(Color.white);
            GL.TexCoord2(textureRect.xMin, textureRect.yMax);
            GL.Vertex3(positionRect.xMin, positionRect.yMax, 0.0f);
            GL.TexCoord2(textureRect.xMax, textureRect.yMax);
            GL.Vertex3(positionRect.xMax, positionRect.yMax, 0.0f);
            GL.TexCoord2(textureRect.xMax, textureRect.yMin);
            GL.Vertex3(positionRect.xMax, positionRect.yMin, 0.0f);
            GL.TexCoord2(textureRect.xMin, textureRect.yMin);
            GL.Vertex3(positionRect.xMin, positionRect.yMin, 0.0f);
            GL.End();
        }
        GL.PopMatrix();
    }

    public IEnumerator Start()
    {
        firstFrame = true;
        //Get Erase effect boundary area
        ScreenRect.x = Dust.renderer.bounds.min.x;
        ScreenRect.y = Dust.renderer.bounds.min.y;
        ScreenRect.width = Dust.renderer.bounds.size.x;
        ScreenRect.height = Dust.renderer.bounds.size.y;

        //Create new render texture for camera target texture
        rt = new RenderTexture(Screen.width, Screen.height, 0, RenderTextureFormat.Default);

        yield return rt.Create();
        Graphics.Blit(tex, rt);
        camera.targetTexture = rt;

        //Set Mask Texture to dust material to Generate Dust erase effect
        Dust.renderer.material.SetTexture("_MaskTex", rt);


    }
```

```
    public void Update()
    {
         newHolePosition = null;
        if (Input.GetMouseButton(0)) //Check if MouseDown
        {
            Vector2 v = camera.ScreenToWorldPoint(Input.mousePosition);
            Rect worldRect = ScreenRect;
            if (worldRect.Contains(v))
            {
                //Get MousePosition for eraser
                newHolePosition = new Vector2(800f * (v.x - worldRect.xMin) / worldRect.width,
600f* (v.y - worldRect.yMin) / worldRect.height);
            }
        }



    }

    public void OnPostRender()
    {
        //Start  It will clear Graphics buffer
        if (firstFrame)
        {
            firstFrame = false;
            GL.Clear(false, true, new Color(0.0f, 0.0f, 0.0f, 0.0f));
        }
        //Generate GL quad according to eraser material texture
        if (newHolePosition != null)
            EraseBrush(new Vector2(800.0f, 600f), newHolePosition.Value);

    }
}
```