

test

by test | Shader id: 2



Your fragmentShader code :
varying vec2 vUv;

```
    // #region Classic Perlin 2D Noise by  
    Stefan Gustavson  
    //
```

```
vec2 fade(vec2 t)
{
    return t*t*t*(t*(t*6.0-15.0)+10.0);
}
```

```
vec4 permute(vec4 x)
{
    return mod(((x*34.0)+1.0)*x, 289.0);
}
```

```
float cnoise(vec2 P)
{
    vec4 Pi = floor(P.xyxy) + vec4(0.0,
0.0, 1.0, 1.0);
    vec4 Pf = fract(P.xyxy) - vec4(0.0,
0.0, 1.0, 1.0);
    Pi = mod(Pi, 289.0); // To avoid
truncation effects in permutation
    vec4 ix = Pi.xzxz;
    vec4 iy = Pi.yyww;
    vec4 fx = Pf.xzxz;
    vec4 fy = Pf.yyww;
    vec4 i = permute(permute(ix) + iy);
```

```

    vec4 gx = 2.0 * fract(i *
0.0243902439) - 1.0; // 1/41 = 0.024...
    vec4 gy = abs(gx) - 0.5;
    vec4 tx = floor(gx + 0.5);
    gx = gx - tx;
    vec2 g00 = vec2(gx.x,gy.x);
    vec2 g10 = vec2(gx.y,gy.y);
    vec2 g01 = vec2(gx.z,gy.z);
    vec2 g11 = vec2(gx.w,gy.w);
    vec4 norm = 1.79284291400159 -
0.85373472095314 * vec4(dot(g00, g00),
dot(g01, g01), dot(g10, g10), dot(g11,
g11));
    g00 *= norm.x;
    g01 *= norm.y;
    g10 *= norm.z;
    g11 *= norm.w;
    float n00 = dot(g00, vec2(fx.x, fy.x));
    float n10 = dot(g10, vec2(fx.y, fy.y));
    float n01 = dot(g01, vec2(fx.z, fy.z));
    float n11 = dot(g11, vec2(fx.w,
fy.w));
    vec2 fade_xy = fade(Pf.xy);

```

```

        vec2 n_x = mix(vec2(n00, n01),
vec2(n10, n11), fade_xy.x);
        float n_xy = mix(n_x.x, n_x.y,
fade_xy.y);
        return 2.3 * n_xy;
    }
    //
    // #endregion

void main()
{
    float strength = step(50,
sin(cnoise(vUv * 3) * 6));
    strength = clamp(strength, 0.0, 1.0);
    vec3 blackColor = vec3(0.98);
    vec3 uvColor = vec3(vUv, 1.0);
    vec3 mixedColor = mix(blackColor,
uvColor, strength);

    gl_FragColor =
vec4(vec3(mixedColor), 1.0);
}

```