

// Procedural Terrain.shader

```
Shader "MinTuts/Procedural Terrain" {
```

```
    Properties {
```

```
        _WaterLimit("Water Limit", Range(0.000001, 0.05)) = 0.01
```

```
        _ShoreLimit("Shore Limit", Range(0.05, 0.1)) = 0.05
```

```
    }
```

```
    SubShader {
```

```
        Pass {
```

```
...
```

```
            #include "UnityCG.cginc"
```

```
            float _WaterLimit;
```

```
            float _ShoreLimit;
```

```
            struct v2f {
```

```
...
```

With **_WaterLimit** and **_ShoreLimit** defined in the Properties section, we must now define them in the **SubShader > Pass > CGPROGRAM** section

This may seem redundant, but it makes sense when you consider that the Properties section exists only to link shader properties to Unity's material inspector

The Properties section does not define anything in the scope of the shader; each line simply says...

"This shader property..."

// Procedural Terrain.shader

```
Shader "MinTuts/Procedural Terrain" {
```

```
    Properties {
```

```
        _WaterLimit("Water Limit", Range(0.000001, 0.05)) = 0.01
```

```
        _ShoreLimit("Shore Limit", Range(0.05, 0.1)) = 0.05
```

```
    }
```

```
    SubShader {
```

```
        Pass {
```

```
...
```

```
            #include "UnityCG.cginc"
```

```
            float _WaterLimit;
```

```
            float _ShoreLimit;
```

```
            struct v2f {
```

```
...
```

With **_WaterLimit** and **_ShoreLimit** defined in the Properties section, we must now define them in the **SubShader > Pass > CGPROGRAM** section

This may seem redundant, but it makes sense when you consider that the Properties section exists only to link shader properties to Unity's material inspector

The Properties section does not define anything in the scope of the shader; each line simply says...

"This shader property...

should be displayed in the inspector using this name,