Step by step guide for integrating Curved World effect into custom shader

1. Include CurvedWorld_Base.cginc file inside vertex shader pass.
   `#include "Assets/VacuumShaders/Curved World/Shaders/cginc/CurvedWorld_Base.cginc"`
   CurvedWorld_Base.cginc file pass will always stay as written above, regardless project setup and folders structure inside.
2. Inside vertex shader use one of two Curved World's vertex transform functions:
   - `inline void V_CW_TransformPoint(inout float4 vertex)` – Transforms only vertex, suitable for unlit shaders.
   - `inline void V_CW_TransformPointAndNormal(inout float4 vertex, inout float3 normal, float4 tangent)` - Transforms vertex and normal, suitable for shaders that require correctly rotated normal for calculating: light, shadow, reflection etc.

   If vertex shader does per-vertex animation, extrude, wind or other per-vertex effects `TransformPoint` function may be used after that.

   Steps 1 and 2 must be used in all vertex shader passes of the shader!

3. (Optional step) If shader uses Fallback shader then it also must be modified for Curved World or can be used one of built-in shaders provided by Curved World:
   - `"Hidden/VacuumShaders/Curved World/VertexLit/Diffuse"` – for opaque shaders.
   - `"Hidden/VacuumShaders/Curved World/VertexLit/Cutout"` – for cutout (alpha test) shaders.
   - `"Hidden/VacuumShaders/Curved World/VertexLit/Transparent"` – for transparent shaders.
4. (Optional step) If shader requires camera Depth and Normal textures for image effects then must be defined custom RenderType or used one of Curved World's built-in RenderTypes:
   - `"RenderType"="CurvedWorld_Opaque"` – for opaque shaders.
   - `"RenderType"="CurvedWorld_TransparentCutout"` – for cutout shaders.

That's all.

Check two example shaders inside Shaders/Example folder:

1. `"Custom/Example_Unlit"`
2. `"Custom/Example_Surface"`

Do not modify original .shader and .cginc files!