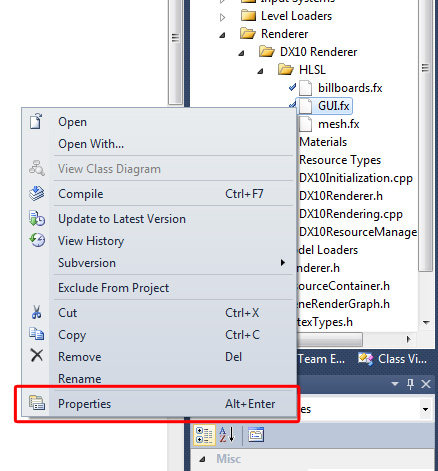
**Debugging HLSL**

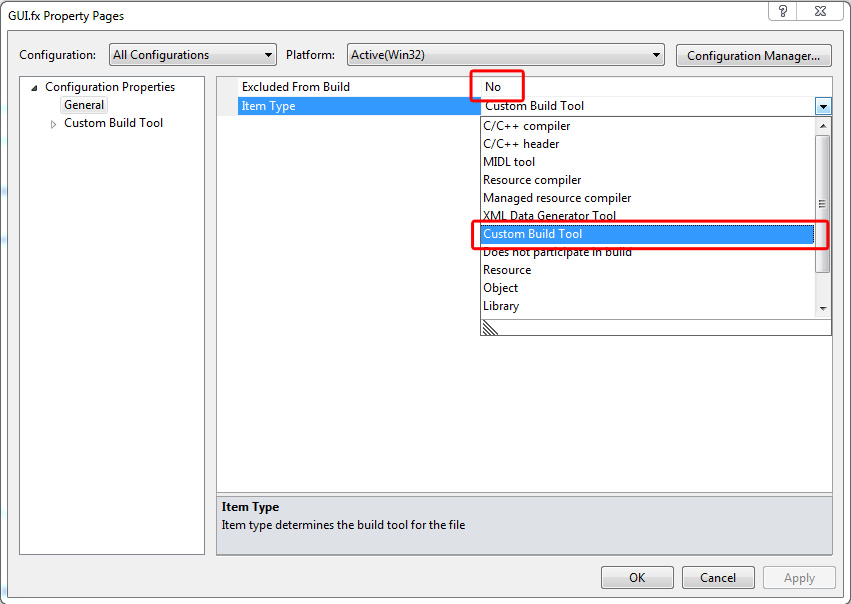
To be able to fully debug shader programs you will need either a shader IDE like FXComposer or a GPU debugging tool like Nvidia Nsight. These are both complex tools and beyond the scope of a quick tutorial but what I can do if provide you a quick guide to aid you in writing shaders directly within Visual Studio. You will not be able to perform any sort of in-depth debugging, but it will help you deal with silly syntax errors. The first thing need is NShader. NShader is a shader syntax highlighting plugin for visual studio and helps with clarity when editing and writing shader programs.

The second thing is to create a custom build step within Visual Studio for your shader programs. This custom build step will use the MS shader compiler to compile your shader programs and notify you of any errors as well as tell you on which lines the errors can be found. To do this, we first select our shader file and right-click then select properties (see figure 1 below).



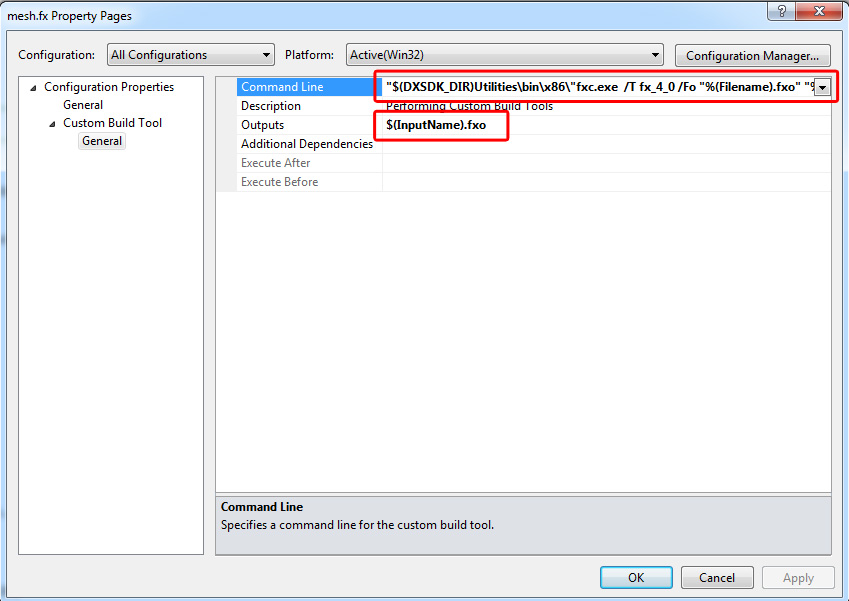
*Figure 1. Shader Code File Properties*

Doing so will bring up the properties windows, the first step is to ensure that the configuration drop down is set to “all configurations”. The select Item Type and choose “Custom Build Tool” from the drop down (see figure 2).



*Figure 2. Enable Custom Build Tool Step*

Click Apply, this will then show the custom build menu tab on the left hand side. Select the tab and you be presented with the following dialog window:



*Figure 3. Set Custom Build Tool Parameters*

Under the general heading, set the command line value to the following:

"$(DXSDK\_DIR)Utilities\bin\x86\"fxc.exe /T fx\_4\_0 /Fo "%(Filename).fxo" "%(FullPath)"

This will mean that every time the shader file is modified and the solution is compiled, the shader file will be compiled using the microsoft FX compiler (FXC). The /T flag specifies the type of shader file being compile (i.e. the HLSL version). the /Fo flag refers to the compiled output file and the %(FullPath) macro refers to the full path of the current shader file.

Also set the Custom Build Tool outputs to: $(filename).fxo , this is the same as specified in the FXC commandline. Click OK, and you are done.

The results of the this process is shown below, all HLSL errors will pop up in the Visual Studio Error Dialog, double clicking on the error will take you to the line of code that caused the error.  
