DX11 Shader Semantics and Annotations

1/Standard Shaders

A/Data convolution annotations

They are in the following format:
typename, annotation typename, type(s). If it applies to every type we use any
for for example float4, string , uiname is
float4 color < string uiname=""name"">;

• float4 / float4 array , bool, , color : Creates a color pin instead of vector pin
• any, bool, visible : Makes the default pin visibility either visible (true) , or in inspector only
  mode (false)
• float4x4 , bool, uvspace: Converts the transform in texture coord space
• any, float, uimin : Minimum value on the pin
• any, float, umax : Maximum value on the pin
• any, float, uistep : Step value on the pin
• bool, bang, bool : Creates a Bang pin (true), or a Toggle pin (false)

B/Transforms

All data in there are float4x4

• WORLD / WORLDTRANSPOSE / WORLDINVERSE /WORLDINVERSETRANSPOSE: World matrix as sent in Transform In pin from shader node
• VIEW / VIEWTRANSPOSE / VIEWINVERSE / VIEWINVERSETRANSPOSE : View matrix from renderer
• PROJECTION / PROJECTIONTRANSPOSE / PROJECTIONINVERSE /
  PROJECTIONINVERSETRANSPOSE : Projection matrix from renderer
• WORLDVIEW : World * View
• WORLDVIEWPROJECTION : World * View * Projection
• VIEWPROJECTION / VIEWPROJECTIONINVERSE /
  VIEWPROJECTIONTRANSPOSE / VIEWPROJECTIONINVERSETRANSPOSE : View
  * Projection

C/Draw information

• BOUNDINGMIN , float3 : Minimum bounding box of the model to be drawn (if available)
• BOUNDINGMAX , float3 : Maximum bounding box of the model to be drawn (if available)
• BOUNDINGSCALE , float3 : Bounding box scale of the model to be drawn (if available)
• OBJUNITTRANS , float4x4 : transform to move the model back into a unit box (-0.5 to 0.5)
• OBJSDFTRANS, float4x4: transform to move the model into a standard sdf space transform (0 to 1)
• DRAWINDEX, int/float: Draw call index for this specific shader
• DRAWCOUNT int/float: Number of draw calls this shader will do (Spreadmax)

D/Render Targets
• BACKBUFFER, RWTexture2D, RWTexture3D, RWStructuredBuffer: When using compute, render target to be drawn (can be texture, volume of buffer). Provided by the renderer
• READBUFFER, Texture2D, Texture3D, StructuredBuffer: Some renderers/plugins might provide a readable buffer to a shader node, so this will be the one
• TARGETSIZE: float2: Render Target size in pixels
• INVTARGETSIZE: float2 Inverse Render Target size
• TARGETSIZE: float3, volume size in voxels, in case of 2d texture Z will be 1
• TARGETSIZE: float4, render target size in xy, inverse in zw
• ELEMENTCOUNT: int, number of elements in buffer type
• BUFFER: RWStructuredBuffer, AppendStructuredBuffer, ConsumeStructuredBuffer: Buffer to write to when using compute and buffer renderer
• VIEWPORTCOUNT: int, number of view/projection/viewport combinations in renderer
• VIEWPORTINDEX: int, index of the currently drawn viewport.

2/Texture FX specific semantics

A/Texture inputs
• INITIAL: Texture input from shader node
• PREVIOUS: Result from previous pass, or same as INITIAL on first pass
• PASSRESULT[n]: Result from a specific pass result, so you can rebind result from pass 1 as input from pass 5 for example
• LASTFRAME: Keeps the target on previous frame and rebinds, uses texture In on first frame, you need to use Technique10 tech <bool persist=true; > to allow texture framedelay.

B/Pass annotations
• mips (bool): Tells that we need to generate mips at the end of the pass
• format (string): Forces render target output to a specific format
• scale (float): Scales the render target size of this factor
• initial (bool): Forces render target size from initial texture input
• tx,ty,tz (int): Thread groups information if using compute shader for this pass