Shaders Specification

2004-09-05

Shader components used for passing parameter to Vertex and Pixel shaders and for other purpose.

Shader components:				
Component name	*Number of floats per component*	*Indexing*	*Only for light pass*	*Description*
LightPos	*1*	*X*	*X*	*Light position in World space*
OSLightPos	*1*	*X*	*X*	*Light position in Object space*
LightIntens	*1*		*X*	*Light intensity for the current light*
InvLightIntens	*1*		*X*	*Inverse Light intensity for the current light (1/LightIntens)*
LightColor	*1*	*X*	*X*	*Light color of the current light source.* *Result = LightColor * MatDiffuseColor*
SpecLightColor	*1*	*X*	*X*	*Specular Light color of the current light source.* *Result = LightSpecularColor * MatSpecularColor*
AmbLightColor	*1*	*X*		*Ambient Light color of the current light source.* *Result = ObjectAmbColor * MatAmbColor*
EngLightColor	*1*	*X*	*X*	*Engine Light color of the current light source.* *Result = WorldColor * LightColor * MatDiffuseColor * ObjectColor;*
EngLeavesLightColor	*1*	*X*	*X*	*Engine Light color specially for plant leaves of the current light source.* *Result = WorldColor * LightColor * MatDiffuseColor * ObjectColor / 1.5f;*
EngAmbColor	*1*	*X*		*Engine Ambient color.* *Res = Eng->`GetWorldAmbientLevel();`*
EngLeavesAmbColor	*1*	*X*		*Engine Ambient color specially for leaves.* *Res = Eng->`GetWorldAmbientLevel() / 1.5f;`*
ObjColor	*1*	*X*		*Color of the current object. Engine should place the values in m_Color member of the object.*

Wave	*1*			*Value evaluated by wave rules:* *Comp 'Wave'* *(* *Type = Sin* *Level = 0.5* *Amp = 0.1* *Phase = 0* *Freq = 0.01* *)* *Here:* *Type – wave type;* *Level – constant level of the wave;* *Amp – Amplitude of the wave;* *Phase – start time phase of the wave;* *Freq – frequency of the wave;*
ObjWaveX, ObjWaveY	*1*			*Waves from the object. For each object we can specify two wave parameters. This parameters currently hardcoded in engine and used for plants bending in X and Y directions respectively..*
FromRE	*1*	*X*		*Value from the current render element. Engine should add appropriate data to the render element.*
FromObject	*1*	*X*		*Value from the current object. Engine should add appropriate data to the object.*
ObjRefrFactor	*1*			*Refraction factor from the current object.*
Time	*1*			*Real time value. Has format "Time fScale". Here fScale – time scale. (for example "Time 0.2")*
Distance	*1*			*Distance from the current object to the camera. Has format "Distance fScale". Here fScale – distance scale. (for example "Distance 0.5")*
VolFogColor	*1*	*X*		*Color of the current fog volume.*
VolFogDensity	*1*			*Density of the current fog volume.*
FogStart	*1*			*Start distance of the global fog.*
FogEnd	*1*			*End distance of the global fog.*
FogRange	*1*			*Range of the global fog.*
CameraAngle	*1*			*Angle of the camera. Format:* *"CameraAngle sSign iInd cOp fValue". Here:* *sSign - "neg" – negative, "pos" – positive;* *iInd – index of the angle (0, 1 or 2);* *iOp - current operation: +, -, * or /.* *fValue – current value for the operation.* *For example "CameraAngle neg 2 * 4" means Value –CameraAngle[2] * 4;*

CameraPos	*1*			*Position of the camera. Format: "CameraPos sSign iInd cOp fValue". Here: sSign - "neg" – negative, "pos" – positive; iInd – index of the angle (0, 1 or 2); iOp - current operation: +, -, * or /. fValue – current value for the operation. For example "CameraPos neg 2 + 3.5" means Value –CameraPos[2] + 3.5;*
OSCameraPos	*1*			*Position of the camera in the object space. Format: "CameraPos sSign iInd cOp fValue". Here: sSign - "neg" – negative, "pos" – positive; iInd – index of the angle (0, 1 or 2); iOp - current operation: +, -, * or /. fValue – current value for the operation. For example "CameraPos neg 2 + 3.5" means Value –CameraPos[2] + 3.5;*
ObjPos	*1*			*Position of the current object. Format: "ObjPos sSign iInd cOp fValue". Here: sSign - "neg" – negative, "pos" – positive; iInd – index of the angle (0, 1 or 2); iOp - current operation: +, -, * or /. fValue – current value for the operation. For example "ObjPos pos 0 – 1.2" means Value ObjPos[0] - 1.2;*
SunColor	*1*	*X*		*Sun color value. Has format "SunColor fScale". Here fScale – color scale. (for example "SunColor[0] 0.2")*
WorldColor	*1*	*X*		*World global color value.*
WorldObjColor	*1*	*X*		*World color value multiplied with current object color.*
ObjVal	*1*	*X*		*Different useful variables in the current object. Engine should place value(s) in m_TempVars member of the object.*
GeomCenter	*1*	*X*		*Center of the current geometry in world space.*
WaterLevel	*1*			*Current water level value. Used for water shaders.*
Bending	*1*			*Bending factor of the current object. Engine should place the value in m_fBending member of the object.*
Bending	*1*			*Bending factor of the current object. Engine should place the value in m_fBending member of the object.*
HalfAngle	*1*	*X*		*Half angle vector for the current light source. Res = Normalize(LightPos + EyePos);*
BumpAmount or BumpScale	*1*			*Shader bump scale value. By default it's 1. Can be changed in shader script.*

