GLSL 1.10.59 : aide mémoire

Types de données

Types réels :	float, vec2, vec3, vec4
Types entiers :	int, ivec2, ivec3, ivec4
Types booléens :	bool, bvec2, bvec3, bvec4
Matrices carrées :	mat2, mat3, mat4
Textures :	sampler1D, sampler2D, sampler3D, samplerCube, sampler1DShadow, sampler2DShadow
Last but not least :	void

Modificateurs de type

Variables globales

uniform	Entrées des *vertex* et *fragment shaders* (lecture seule) ;
attribute	Entrées par sommet du *vertex shader* (lecture seule) ;
varying	Sorties du *vertex shader* (lecture/écriture), interpolées et fournies en entrée du *fragment shader* (lecture seule) ;
const	Constante connue dès la compilation (lecture seule).

Paramètres de fonction

in	initialisé à l'entrée, non modifié en sortie (défaut) ;
out	modifié en sortie, non initialisé à l'entrée ;
inout	initialisé à l'entrée, modifié en sortie ;
const	valeur constante (non modifiable).

Composantes de vecteurs

x, y, z, w ou r, g, b, a ou s, t, p, q

Fonctions prédéfinies

Légende :
vec : vec2 — vec3 — vec4
mat : mat2 — mat3 — mat4
ivec : ivec2 — ivec3 — ivec4
bvec : bvec2 — bvec3 — bvec4
tGen : float — vec2 — vec3 — vec4

Fonctions trigonométriques

tGen sin(tGen)
tGen cos(tGen)
tGen tan(tGen)
tGen asin(tGen)
tGen acos(tGen)
tGen atan(tGen, tGen)
tGen atan(tGen)
tGen radians(tGen)
tGen degrees(tGen)

Fonctions exponentielles

tGen pow(tGen, tGen)
tGen exp(tGen)
tGen log(tGen)
tGen exp2(tGen)
tGen log2(tGen)
tGen sqrt(tGen)
tGen inversesqrt(tGen)

Fonctions geométriques

vec3 cross(vec3, vec3)
float distance(tGen, tGen)
float dot(tGen, tGen)
tGen faceforward(tGen V,tGen I,tGen N)
float length(tGen)
tGen normalize(tGen)
tGen reflect(tGen I, tGen N)
tGen refract(tGen I,tGen N,float eta)

Fonctions communes

tGen abs(tGen)
tGen ceil(tGen)
tGen clamp(tGen, tGen, tGen)
tGen clamp(tGen, float, float)
tGen floor(tGen)
tGen fract(tGen)
tGen max(tGen, tGen)
tGen max(tGen, float)
tGen min(tGen, tGen)
tGen min(tGen, float)
tGen mix(tGen, tGen, tGen)
tGen mix(tGen, tGen, float)
tGen mod(tGen, tGen)
tGen mod(tGen, float)
tGen sign(tGen)
tGen smoothstep(tGen, tGen, tGen)
tGen smoothstep(float, float, tGen)
tGen step(tGen, tGen)
tGen step(float, tGen)

Relations vectorielles

bool all(bvec)
bool any(bvec)
bvec equal(vec, vec)
bvec equal(ivec, ivec)
bvec equal(bvec, bvec)
bvec greaterThan(vec, vec)
bvec greaterThan(ivec, ivec)
bvec greaterThanEqual(vec, vec)
bvec greaterThanEqual(ivec, ivec)
bvec lessThan(vec, vec)
bvec lessThan(ivec, ivec)
bvec lessThanEqual(vec, vec)
bvec lessThanEqual(ivec, ivec)
bvec not(bvec)
bvec notEqual(vec, vec)
bvec notEqual(ivec, ivec)
bvec notEqual(bvec, bvec)

Fonctions matricielles

mat matrixCompMult(mat, mat)
mat transpose(mat)

Echantillonnage de textures

vec4 texture1D(sampler1D, float [,float bias])
vec4 texture1DProj(sampler1D, vec2 [,float bias])
vec4 texture1DProj(sampler1D, vec4 [,float bias])
vec4 texture2D(sampler2D, vec2 [,float bias])
vec4 texture2DProj(sampler2D, vec3 [,float bias])
vec4 texture2DProj(sampler2D, vec4 [,float bias])
vec4 texture3D(sampler3D, vec3 [,float bias])
vec4 texture3DProj(sampler3D, vec4 [,float bias])
vec4 textureCube(samplerCube, vec3 [,float bias])
vec4 shadow1D(sampler1DShadow, vec3 [,float bias])
vec4 shadow2D(sampler2DShadow, vec3 [,float bias])
vec4 shadow1DProj(sampler1DShadow,vec4 [,float bias])
vec4 shadow2DProj(sampler2DShadow,vec4 [,float bias])

Echantillonnage de texture avec niveau de détail (LOD)

(A n'utiliser que dans le *vertex shader*)

vec4 texture1DLod(sampler1D, float, float lod)
vec4 texture1DProjLod(sampler1D, vec2, float lod)
vec4 texture1DProjLod(sampler1D, vec4, float lod)
vec4 texture2DLod(sampler2D, vec2, float lod)
vec4 texture2DProjLod(sampler2D, vec3, float lod)
vec4 texture2DProjLod(sampler2D, vec4, float lod)
vec4 texture3DProjLod(sampler3D, vec4, float lod)
vec4 textureCubeLod(samplerCube, vec3, float lod)
vec4 shadow1DLod(sampler1DShadow, vec3, float lod)
vec4 shadow2DLod(sampler2DShadow, vec3, float lod)
vec4 shadow1DProjLod(sampler1DShadow,vec4,float lod)
vec4 shadow2DProjLod(sampler2DShadow,vec4,float lod)

Traitement des fragments

(A n'utiliser que dans le *fragment shader*)

tGen dFdx(tGen)
tGen dFdy(tGen)
tGen fwidth(tGen)

Variables spéciales du *vertex shader*

Sortie (accès en lecture/écriture)

vec4 gl_Position	position du sommet (à modifier impérativement) ;
float gl_PointSize	taille du point en pixel ;
vec4 gl_ClipVertex	coordonnées du sommet par rapport aux plans de clipping.

Variables spéciales du *fragment shader*

Sortie (accès en lecture/écriture)

vec4 gl_FragColor	couleur du fragment de sortie ;
vec4 gl_FragData[]	données associées au fragment ;
float gl_FragDepth	profondeur du fragment.

Entrées (accès en lecture seule)

vec4 gl_FragCoord	coordonnées du fragment
bool gl_FrontFacing	indique si le fragment est sur le dessus du polygone.

Shaders minimaux avec three.js

Vertex shader

Calcul de la position finale du sommet en effectuant les multiplications matricielles de rigueur.

```
void main() {
  gl_Position = projectionMatrix * modelViewMatrix *
      vec4(position,1);
}
```

Fragment shader

Calcul de la couleur finale d'un fragment, ici blanc.

```
void main() {
  gl_FragColor = vec4(1.0,1.0,1.0,1.0);
}
```

Mis en page par Jean-Yves Didier, le 7/3/2013.
Version originale : Mike Weiblen (`http://mew.cx/`)