# 顶点着色器：

## 全局变量：

## gl\_Position:顶点的位置

## gl\_PointSize：顶点的大小

# 数据类型

## vec4表述由四个浮点数组成的矢量

## float：表示浮点数

## Float32Array

写入缓冲区对象的顶点数据类型

var vertices = new Float32Array([

0.0, 0.5, -0.5, -0.5, 0.5, -0.5

]);

与之类似的还有[u]int[8,16,32]array等等

产生的数据支持get,set，而不支持push和pop操作。

## Mat4 矩阵类型（作者自定义类型）

var xformMatrix = new Float32Array([

cosB, sinB, 0.0, 0.0,

-sinB, cosB, 0.0, 0.0,

0.0, 0.0, 1.0, 0.0,

0.0, 0.0, 0.0, 1.0

]);

实际上底层还是floatarray

# 片元着色器

实际上就是在屏幕上的一个像素，片元包括像素的位置，颜色和其他信息

## 全局变量：

## gl\_FragColor：片元着色器的颜色

## texture2D ：片元着色器-纹理

texture2D(u\_Sampler, v\_TexCoord)

第一个参数为采样器，采样器在着色器内部声明，由外部传入

使用gl.uniform1i 函数将纹理单元赋值给采样器即可

第二个参数为纹理坐标

## sampler2D 采样器

# 常用函数：

## Gl. drawArrays(mode,first,count)---书82页

Mode表示指定绘制的方式，包括，line,points,lines,line\_loop,triangle等等

First指定从哪个点开始绘制

Count表示要绘制点的数量

绘制三角带时，第二个三角形的绘制顺序是(v2,v1,v3)，为的是保持逆时针顺序。

理解：v2是第一个三角形的终点，一定是第二个三角形的起点；保持逆时针，因此形成这个顺序。

三角扇：第二个三角形是(v0,v2,v3)-三角扇以初始点为中心形成扇形，会回到初始点

理解：连续画三角形，第一次绘制结束时，终点是v0，那么第二个三角形的起点是v0,保持逆时针，则形成这个三角形

## gl.getAttribLocation(gl.program, 'a\_Position')

获取顶点着色器存储attribute变量的地址

第一个参数是gl程序，调用initshader后可使用，第二个参数为顶点着色器的变量名

## gl.vertexAttrib3f(a\_Position, 0.8, 0.0, 0.0);

向顶点着色器attribute变量传输顶点坐标数据

函数名称示意：顶点着色器-attribute变量 三维的float数据

类似还有：gl.vertexAttrib1,2,3,4f

## gl.getUniformLocation(gl.program, 'u\_FragColor');

获取片元着色器变量的地址

## gl.uniform4f(u\_FragColor, rgba[0], rgba[1], rgba[2], rgba[3]);向片元着色器变量赋值

## gl.bindBuffer(gl.ARRAY\_BUFFER, vertexBuffer);

绑定缓冲区对象到webgl系统中，

第一个参数可以是gl.array\_buffer，可以是gl.elemnet\_ array\_buffer，可以是buffer

gl.array\_buffer：表示缓冲区对象包含了顶点数据

gl.elemnet\_ array\_buffer：表示缓冲区对象包含了顶点的索引值

## gl.createBuffer();

创建缓冲区对象，可将多个矢量信息存储到缓冲区对象中；缓冲区对象是独立于webgl基层系统的一块区域

## gl.bufferData(gl.ARRAY\_BUFFER, vertices, gl.STATIC\_DRAW);

将数据传入到缓冲区对象中-即传入到webgl系统中

将第二个参数的数据写入到第一个参数中

第三个参数表示值写入一次

第三个参数可选值

gl.STATIC\_DRAW：只向缓冲区对象写入一次数据，绘制多次

gl.STREAM\_DRAW：只向缓冲区对象写入一次数据，绘制若干次

gl.DYNAMIC\_DRAW：向缓冲区对象写入多此数据，绘制多次

## gl.vertexAttribPointer(a\_Position, 2, gl.FLOAT, false, 0, 0);

将缓冲区对象分配给a\_position变量

（locatin,size,type,normalized,stride,offset）

第一个参数：指定待分配attribute变量的存储位置

第二个参数，取出数据得分量个数，每次取两个，在绘制时，会默认补上

第三个参数：数据类型

第四个参数：是否将非浮点数据规划到[-1,1]

第五个参数：相邻两个顶点之间的字节数

第六个参数：偏移量

## gl.enableVertexAttribArray(a\_Position);

链接a\_position变量与分配给他的缓冲区对象

开启attribute变量，与之相对应的是diable，关闭分配

## gl. diableVertexAttribArray(a\_Position);

## gl.createTexture()

创建纹理对象，对应的还有删除纹理对象

## gl.uniformMatrix4fv (u\_xformMatrix, false, xformMatrix);

第一个参数为着色器中变量地址

第三个参数为值

## gl.deleteTexture(texture)

## gl.pixelStorei(gl.UNPACK\_FLIP\_Y\_WEBGL, 1)

对纹理图像进行Y轴反转，此时还未加载图像，

第一个参数：以指定的方式处理加载的图像

gl.UNPACK\_FLIP\_Y\_WEBGL：Y轴反转，默认为false

，另一个可选值为图像的每个分量乘A，默认为false

第二个参数为指定0或者非0，暂未理解作用

## gl.activeTexture(gl.TEXTURE0);

开启0号单元，webgl中用纹理单元来管理纹理对象，至少8个

## gl.bindTexture(gl.TEXTURE\_2D, texture);

// 向target绑定纹理对象，类似于缓冲区，指定纹理对象的类型2d,和立方体纹理

第一个参数可选值gl.TEXTURE\_2D,gl.TEXTURE\_CUBE\_MAP

第二个参数为纹理变量的地址

## gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_MIN\_FILTER, gl.LINEAR);

// 设置纹理参数，第二个参数，纹理放大/缩小，水平填充/竖直填充，第二个参数表示填充时的采样方式，类似于div中放置图片

第二个参数可选值：

|  |  |  |
| --- | --- | --- |
| 纹理参数 | 描述 | 默认值 |
| gl.TEXTURE\_MIN\_FILTER | 纹理放大 | gl.LINEAR， |
| gl.TEXTURE\_MAG\_FILTER | 纹理缩小 | gl.NEARST\_MIPMAP\_LINEAR |
| gl.TEXTURE\_WRAP\_S | 处理水平填充 | gl.REPAET |
| gl.TEXTURE\_WRAP\_T | 处理垂直填充 | gl.REPAET |

1,2两种纹理参数可互相交换

3,4两种可选的纹理值还包括GL.MIRRORED\_REPEAT,GL.CLAMP\_TO\_EDGE

## gl.texImage2D(gl.TEXTURE\_2D, 0, gl.RGB, gl.RGB, gl.UNSIGNED\_BYTE, image);

// 设置纹理图像，第二个参数，金字塔纹理使用；第三个参数，图像内部的颜色格式，第四个参数，纹理数据的格式，第五个参数，纹理数据类型

纹理数据格式可选参数

Gl.rgb

gl.rgba,gl.ALPHA,等

纹理数据的数据格式

## gl.uniform1i(u\_Sampler, 0);

// 将0号纹理赋值给着色器

## Gl.drawElements()

第一个参数指定绘制方式：points,gl.lines,line\_strip,line\_loop,triangle, triangle\_strip, triangle\_fan

第二个参数：count

第三个参数：指定索引数据类型 unsigned\_byte,unsigned\_short

第四个参数：offset

# 常用变量

## Attribute---顶点着色器使用

传输与顶点相关的数据---坐标信息

在顶点着色器中声明该变量，将该变量赋值给gl\_position

向attribute传输数据

## Uniform—顶点片元都可以使用

传输与顶点无关的数据---属性信息

在片元着色器中准备uniform变量

变量赋值给gl\_fragColor

Js传输数据

uniform变量可以在顶点着色器与片元着色器中，且必须是全局的。与顶点无关，对所有顶点相同如变换矩阵。如果在顶点与片源着色器中使用了同名的uniform变量，那么就会被两种着色器共享。

## Varying

在顶点着色器中中声明varying color接收变量，并在片元着色器中声明相同的变量名称来接收

任务为从顶点着色器向片源着色器传输数据,必须在两种着色器中声明同名的，同类型的varying变量。顶点着色器中赋值给varying变量的值经过内插传给了片元着色器的varying变量。

# 第六章 着色器

## 数据类型

### 数值类型

整数，浮点数，无小数点即为证书

Int float

### 布尔值类型

bool

不支持字符串类型

### 矢量和矩阵

矢量：

vec2,vec3,vec4 浮点数元素的矢量

ivec2,ivec3,ivec4 整形矢量

bvec2,bvec3,bvec4 布尔值矢量

矩阵：

Mat2,mat3,mat4 浮点数矩阵

## 矢量构造函数

即为类型本身vec3(1.0,1.0,1.0)

多个矢量合成一个矢量

Vec4 v4b=vec4(vec2,vec4)先用vec的元素进行填充，如未填满，就继续用第二个参数中的元素

自动补全：vec3（1.0） 结果为(1.0,1.0,1.0)

## 矩阵构造函数

Mat4()构造

运用矢量构造，按列主序构造如(1,2) (3,4) 构造成为

（

1, 2

3 ,4）

使用矢量构造矩阵（1,2,3,4）

构造mat2为

1,2

3,4

数值和矢量同时作为参数：以列主序构造

单值构造：mat4(1.0)---主对角线为1.0的四阶矩阵

与矢量构造函数类似，如果传入参数大于1，又未达到目标个数，则会报错

## 访问元素

X,y,z,w 顶点坐标分量

R,g,b,a 颜色分量

S,t,q,p 纹理坐标分量

任何矢量的x,r,s都会返回第一个分量，y,g,t返回第二个分量（因为矢量只有一行）

同一个集合的多个分量名共同置于点运算符后，可以从矢量中同时抽取多个分量

Vec3(1,2,3) v3.xy=(1,2)

可以使用点访问，可以使用[]进行访问

M[2][3] 表示访问第二列的第三个元素，列主序的体现

## 运算符

常规运算符中，除去大小不可使用之外，其他基本都相似

矩阵左乘矢量和右乘矢量—按理是不可交换的 右乘按3×3 \*3×1计算

左乘按1×3 \*3×3计算

## 结构体

Struct person{

Vec color

} person1 person类型的变量 person1

构造函数参数的顺序必须与定义的顺序一致

## 数组

不支持pop和push操作

Float floatArray[4] 声明4个浮点数元素的数组

Vec4 vec4Array[2]声明含有2个vec对象的数组

数组长度必须是大于0的整型常量表达式—即不可动态声明数组的长度

Int size=4, Float floatArray[size] 错误

Const Int size=4, Float floatArray[size] 正确

## 取样器

取样器只能是uniform类型的变量

Uniform sampler2D u\_Sampler

Uniform samplerCube u\_Sampler

## 流程控制语句

For if-else continu break discard

## 函数

规范声明：定义在调用之后，则需要在调用之前进行声明

## 参数限定词

|  |  |  |
| --- | --- | --- |
| 类别 | 规则 | 描述 |
| In（默认值） | 向函数中传入值 | 函数中可以修改参数值 |
| Const in | 向函数中传入值 | 函数中不可以修改参数值 |
| Out | 在函数中赋值，并传出 | 外部可能不声明 |
| inout | 传入函数在函数中赋值，并传出 |  |

## 内置函数

|  |  |
| --- | --- |
| 类别 | 使用 |
| 角度函数 |  |
| 三角函数 |  |
| 指数函数 |  |
| 通用函数 |  |
| 几何函数 |  |
| 矩阵函数 |  |
| 矢量函数 |  |
| 纹理查询函数 |  |

## 存储限定符

Const 常量

Uniform：可以是除了数组和结构体之外的任意类型，顶点着色器中和片元着色器中声明了相同的uniform变量，则该值会被顶点和片元着色器共用

Attribute：只能在顶点着色器中使用，只能是vec234,或者Mat234类型

Varying:从顶点着色器向片元着色器传数据的限定符

## 精度限定字

Lowp

Medium

Highp

## 预处理指令

宏定义等 #开头