通过使用glVertexAttribPointer，我们能够指定顶点数组缓冲内容的属性布局。在顶点数组缓冲中，我们对属性进行了交错(Interleave)处理，也就是说，我们将每一个顶点的位置、发现和/或纹理坐标紧密放置在一起。既然我们现在已经对缓冲有了更多的了解，我们可以采取另一种方式。

我们可以做的是，将每一种属性类型的向量数据打包(Batch)为一个大的区块，而不是对它们进行交错储存。与交错布局123123123123不同，我们将采用分批(Batched)的方式111122223333。

当从文件中加载顶点数据的时候，你通常获取到的是一个位置数组、一个法线数组和/或一个纹理坐标数组。我们需要花点力气才能将这些数组转化为一个大的交错数据数组。使用分批的方式会是更简单的解决方案，我们可以很容易使用glBufferSubData函数实现：

loat positions[] = { ... };

float normals[] = { ... };

float tex[] = { ... };

// 填充缓冲

glBufferSubData(GL\_ARRAY\_BUFFER, 0, sizeof(positions), &positions);

glBufferSubData(GL\_ARRAY\_BUFFER, sizeof(positions), sizeof(normals), &normals);

glBufferSubData(GL\_ARRAY\_BUFFER, sizeof(positions) + sizeof(normals), sizeof(tex), &tex);

glVertexAttribPointer(0, 3, GL\_FLOAT, GL\_FALSE, 3 \* sizeof(float), 0);

glVertexAttribPointer(1, 3, GL\_FLOAT, GL\_FALSE, 3 \* sizeof(float), (void\*)(sizeof(positions)));

glVertexAttribPointer(

2, 2, GL\_FLOAT, GL\_FALSE, 2 \* sizeof(float), (void\*)(sizeof(positions) + sizeof(normals)));

当你的缓冲已经填充好数据之后，你可能会想与其它的缓冲共享其中的数据，或者想要将缓冲的内容复制到另一个缓冲当中。glCopyBufferSubData能够让我们相对容易地从一个缓冲中复制数据到另一个缓冲中。这个函数的原型如下：

void glCopyBufferSubData(GLenum readtarget, GLenum writetarget, GLintptr readoffset,

GLintptr writeoffset, GLsizeiptr size);