# 6.3 索引和索引缓冲

由于索引要由GPU访问，所以它们必须放在一个特定的资源容器中，该容器称为索引缓冲（index buffer）。创建索引缓冲的过程与创建顶点缓冲的过程非常相似，只不过索引缓冲存储的是索引而非顶点。所以，这里不再赘述之前讨论过的内容，我们直接给出一个创建索引缓冲区的示例：

UINT indices [24] = {

0, 1, 2, // Triangle 0

0, 2, 3, // Triangle 1

0, 3, 4, // Triangle 2

0, 4, 5, // Triangle 3

0, 5, 6, // Triangle 4

0, 6, 7, // Triangle 5

0, 7, 8, // Triangle 6

0, 8, 1 // Triangle 7

} ;

// 要创建的索引的描述

D3D11\_BUFFER\_DESC ibd;

ibd.Usage = D3D11\_USAGE\_IMMUTABLE;

ibd.ByteWidth = sizeof(UINT) \* 24;

ibd.BindFlags = D3D11\_BIND\_INDEX\_BUFFER;

ibd.CPUAccessFlags = 0;

ibd.MiscFlags = 0;

ibd.StructureByteStride = 0;

// 设定用于初始化索引缓冲的数据

D3D11\_SUBRESOURCE\_DATA iinitData;

iinitData.pSysMem = indices;

// 创建索引缓冲

ID3D11Buffer\* mIB;

HR(md3dDevice->CreateBuffer(&ibd, &iinitData, &mIB));

与顶点缓冲区相同，所有的Direct3D资源在使用之前都必须先绑定到管线上。我们使用**ID3D11DeviceContext::IASetIndexBuffer**方法将一个索引缓冲区绑定到输入装配阶段。下面是一个例子：

md3dImmediateContext->IASetIndexBuffer(mIB, DXGI\_FORMAT\_R32\_UINT, 0);

第2个参数表示索引格式。在本例中，我们使用的是32位无符号整数（DWORD）；所以，该参数设为DXGI\_FORMAT\_R32\_UINT。如果你希望节约一些内存，不需要这大的取值范围，那么可以改用16位无符号整数。还要注意的是，在**IASetIndexBuffer**方法中指定的格式必须与**D3D11\_BUFFER\_DESC::ByteWidth**数据成员指定的字节长度一致，否则会出现问题。索引缓冲区只支持**DXGI\_FORMAT\_R16\_UINT**和**DXGI\_FORMAT\_R32\_UINT**两种格式。第3个参数是一个偏移值，它表示从索引缓冲区的起始位置开始、到输入装配时实际读取数据的位置之间的字节长度。如果希望跳过索引缓冲区前面的一部分数据，那么可以使用该参数。

最后，当使用索引时，我们必须用**DrawIndexed**方法代替**Draw**方法：

void ID3D11DeviceContext::DrawIndexed(

UINT IndexCount,

UINT StartIndexLocation,

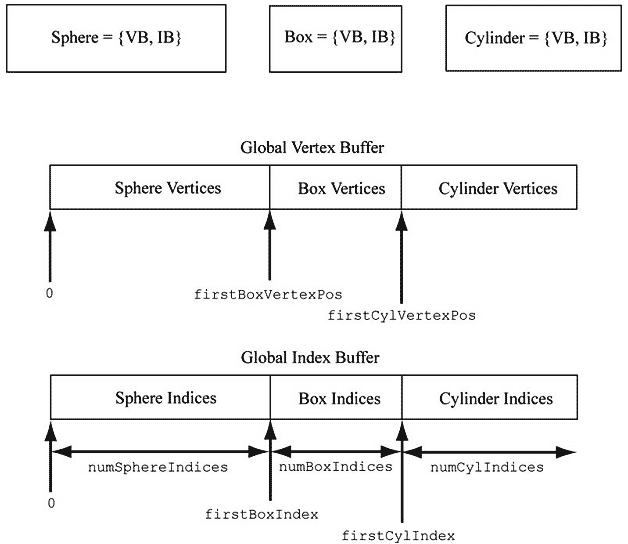
INT BaseVertexLocation);

**1．IndexCount**：在当前绘图操作中使用的索引的数量。在一次绘图操作中不一定使用索引缓冲区中的全部索引；也就是说，我们可以绘制索引的一个连续子集。

**2．StartIndexLocation**：指定从索引缓冲区的哪个位置开始读取索引数据。

**3．BaseVertexLocation**：在绘图调用中与索引相加的一个整数。

我们通过分析如下情景来解释些参数。假设有三个物体：一个球体、一个立方体和一个圆柱体。每个物体都有它自己的顶点缓冲和索引缓冲。在每个独立的索引缓冲中的索引都与各自独立的顶点缓冲对应。现在，我们把这三个物体的顶点和索引数据合并到一个全局的顶点缓冲和一个全局的索引缓冲中，如图6.3所示（假如有许多小顶点缓冲和索引缓冲，并且它们可以很容易的合并，那么合并它们可以带来性能提升）。在合并之后，索引就不再正确了（因为这些索引是与各自独立的顶点缓冲区对应的，它们与全局顶点缓冲区没有对应关系）；所以，这些索引必须被重新计算，使它们与全局顶点缓冲区建立正确的对应关系。

****

**图6.3 将多个顶点缓冲合并为一个大的顶点缓冲，将多个索引缓冲合并为一个大的索引缓冲。**

假设原先的立方体索引为0、1、...、numBoxVertices-1，通过个索引可以遍历立方体的所有顶点。在合并之后，索引应该变为firstBoxVertexPos、firstBoxVertexPos+1、...、firstBoxVertexPos+ numBoxVertices -1。所以，要更新索引，我们必须将firstBoxVertexPos与立方体的每个索引相加。而且，我们必须将firstCylVertexPos与圆柱体的每个索引相加。注意，球体的索引不需要修改（因为球体的顶点位置为0）。通常，只要将一个物体在全局顶点缓冲区中的第一个顶点位置与原索引相加就可以得到该物体的新索引值。所以，只要给出物体在全局顶点缓冲中的第一个顶点的位置（例如，BaseVertexLocation），Direct3D就可以在绘图调用中为我们重新计算索引。我们可以使用以下3条语句依次绘制球体、立方体和圆柱体：

md3dImmediateContext->DrawIndexed(numSphereIndices, 0, 0);

md3dImmediateContex->DrawIndexed(numBoxIndices, firstBoxIndex, firstBoxVertexPos);

md3dImmediateContex->DrawIndexed(numCylIndices, firstCylIndex, firstCylVertexPos);

后面的“Shape”示例程序就用到了这个技术。