# 4.6 小结

**1．**Direct3D可以被视为程序员和图形硬件之间的一个中介。例如，程序员调用Direct3D函数将资源视图绑定到硬件渲染管线、设定渲染管线的输出并绘制3D几何体。

**2．**在Direct3D 11中，一个支持Direct3D 11的图形设备必须支持 Direct3D 11规定的整个功能集合以及少量的额外功能。

**3．**组件对象模型（COM）技术使DirectX独立于任何编程语言，并具有版本向后兼容的特性。Direct3D程序员不必知道COM的实现细节及工作方式；只需要知道如何获取和释放COM接口即可。

**4．**1D纹理如同一维数据元素数组，2D纹理如同二维数据元素数组，3D纹理如同三维数据元素数组。纹理元素的格式由**DXGI\_FORMAT**枚举类型成员描述。纹理通常用于存储图像数据，但是也可以用于存储其他数据，比如深度信息（例如，深度缓冲区）。GPU可以在纹理上执行特殊运算，比如过滤器和多重采样。

**5．**在Direct3D中，资源不能被直接绑定到一个管线阶段；我们只能把与资源关联的资源视图绑定到不同的管线阶段。我们可以为一个资源创建多个不同的视图。通过这一方式，一个资源可以被绑定到多个不同的渲染管线阶段。如果在创建资源时使用的是弱类型格式，那么在为该资源创建视图时必须指定明确的类型。

**6．ID3D11Device**和**ID3D11DeviceContext**接口可以被视为物理图形设备硬件的软控制器；也就是，我们可以通过这些接口与硬件进行交互。**ID3D11Device**接口负责检查硬件支持的功能、分配资源。**ID3D11DeviceContext**接口负责设置渲染状态，将资源绑定到图形管线，发送渲染指令。

**7．**为了避免动画出现闪烁，最好是将整个帧绘制到一个叫做后台缓冲区的离屏纹理中。当整个屏幕绘制到后台缓冲区之后，它就会以一个完整帧的形式呈现在屏幕上，通过这种方式，观察者就不会觉察到图像的绘制过程了。当帧绘制到后台缓冲区之后，后台缓冲和前台缓冲就会发生互换：后台变前台，前台变后台。交换两者的过程叫做呈现（presenting）。前台缓冲和后台缓冲构成一个交换链，由IDXGISwapChain接口表示，使用两个缓冲被称为双缓冲。

8．对于屏幕上不透明的物体来说，离相机近的点会遮挡后面的点。深度缓冲就是一种判断哪个点离相机近的技术。通过这一方式，我们就无需关心对象的绘制顺序。

9．性能计数器是一种高精度计时器，它为测量微小的时间差提供了准确无误的计时测量方法，比如帧之间的时间间隔。性能计时器采用的时间单位叫做计数。**QueryPerformanceFrequency**函数用于输出性能计时器每秒的计数值，这个值的单位可以从计数转换为秒。我们可以通过**QueryPerformanceCounter**函数获取性能计时器的当前时间。

10．我们通过累计某一时间段Δ*t*内的帧数来计算FPS（frames per second，每秒帧数）。设*n*为时间段Δ*t*中的帧数；在一段时间中的平均每秒帧数为*fps*avg＝*n*/Δ*t*。帧速率会对性能评定产生误导；帧时间是更有效的信息。以秒为单位的帧时间等于帧速率的倒数，即1/*fps*avg。

11．示例框架用于为本书的所有演示程序提供统一的编程接口。这些代码保存在d3dUtil.h、d3dApp.h和d3dApp.cpp文件中，它们封装了每个应用程序必须实现的标准初始化代码。通过封装些代码，可以使示例程序更专注于所要演示的技术。

12．当以调试模式生成程序时，我们可以使用D3D11\_CREATE\_DEVICE\_DEBUG标志值创建Direct3D设备来启用调试层。在指定了调试标志值后，Direct3D会把调试信息发送到VC++的输出窗口。另外，当以调试模式生成程序时，我们应使用D3DX库的调试版本（即，d3dx11d.lib）。