# Basic对象

里面有很多概念，我们逐一来讲解，可以先看看这里

[DX12基本介绍 - 知乎](https://zhuanlan.zhihu.com/p/634854048)

# DXGI对象

但是上面的介绍还少了一些具体的描述，

首先是 DXGI，这是个什么？他所包含的对象都有哪些？

[DXGI 概述 - Win32 apps | Microsoft Learn](https://learn.microsoft.com/zh-cn/windows/win32/direct3ddxgi/d3d10-graphics-programming-guide-dxgi)

DXGI简单来讲，就是最终驱动硬件的接口，D3D是负责图形绘制（也就是画好一张图），但是不负责图形的输出（也就是发信号到显卡，显卡再输出到显示器），而DXGI负责这一块内容

A diagram of a computer system

AI-generated content may be incorrect.

DXGI为我们这些硬件抽象出来了一个软件对象的，显卡就是IDXGIAdapter，显示器就是IDXGIOutput，一般，一个显卡可以输出到多个显示器。

而在写代码的过程中，枚举这些对象的就是 IDXGIFactory。

虽然 IDXGIAdapter 是抽象出来的显卡对象，但是他只是这个显卡的描述，并不包含直接调用显卡的方法，真正代表显卡的是 IDXGIDevice，同时最重要的，IDXGIDevice就是绘制图形的API接口，你可以像理解文件系统一样看这个事情，IDXGIAdapter是文件头，而IDXGIDevice是真正的文件。

[IDXGIDevice (dxgi.h) - Win32 apps | Microsoft Learn](https://learn.microsoft.com/zh-cn/windows/win32/api/DXGI/nn-dxgi-idxgidevice)

通过 IDXGIAdapter 可以创建出 IDXGIDevice。

另外一个值得说明的就是 SwapChain，他也是 DXGI 的部分，只不过他的创建需要一个command queue对象。

# D3D对象

由 d3d device 创建出来或者间接创建出来的对象，都是 D3D 对象

<https://learn.microsoft.com/en-us/windows/win32/direct3d12/command-queues-and-command-lists>

先粗略去阅读下上面的文章，看看基本的流程（上面的帖子相当于d3d 的百科了。）

## Command queue， Allocator， List

Command list就是一堆gpu指令的集合，但是，他并不是存储gpu指令的地方，你应该把它看成一个接口，真正存储这些gpu指令的是Allocator，command list的作用是提供接口让你把gpu指令record到Allocator里面，Allocator本质上是一段连续内存，当调用 ExecuteCommandList的时候，就会从Allocator中拷贝这些指令到对应的显存然后执行。所以，如果你今天的gpu比较拉跨，cpu这边生成了一大堆gpu指令，但是gpu还没执行完前面的指令，你就相当于需要弄多几个 Allocator 把指令缓存起来，你在调用ExecuteCommandList只是告诉Command queue把这些缓存拼起来，等gpu慢慢执行过来。这也是为什么gpu还没执行完一个allocator上的指令的时候，你可以随意调用command list的reset接口，因为command list并不是一个真正的gpu指令对象，他只是一个接口，真正的数据在allocator，你在它reset的时候传入新的allocator即可。

为什么要这样的设计，主要就是为了让cpu和gpu可以不同步，可以充分利用cpu的多核性能，gpu那边也可以尽情优化自己的指令。

## Fence

试想一个情况，你想要让gpu算一个东西，然后你cpu等gpu算好了再去读取并使用，那么你就要让cpu等待一下gpu，那么fence就是这么个用途，一开始你在cpu端设置fence=0，然后你在command list的尾巴插入令 fence = 1的指令，那么cpu只有当执行完这个commandlist后，才会将fence赋值为1，而这期间，你cpu就可以不断去轮询fence的值，只要还没达到cpu就一直等。听着是不是很像多线程里面的信号量

## View

在gpu中，会经常有各种buffer，texture（我们称他们为resource），这个时候，需要你告诉gpu这个buffer是什么，怎么读取他的格式，怎么解析它，而这个告诉gpu这个事情的，就是D3D12\_RESOURCE\_DESC，而一个resource可以一会被这样解析，一会被那样解析（比如一个贴图包含了normal以及height信息），所以这个resource可以整合两个不同的desc，而将一个resource和desc绑定在一起的叫view，也就是一个view就是一个resource+desc的意思。另外一点，就是desc通常是需要用一个heap来存储，这个heap