## Staging buffer;Transform queue

finish

## Staging buffer;mangae memory buffer

You can either implement such an allocator yourself, or use the [VulkanMemoryAllocator](https://github.com/GPUOpen-LibrariesAndSDKs/VulkanMemoryAllocator) library provided by the GPUOpen initiative. However, for this tutorial it's okay to use a separate allocation for every resource, because we won't come close to hitting any of these limits for now.

## Index buffer；go further in managing memory buffer

The previous chapter already mentioned that you should allocate multiple resources like buffers from a single memory allocation, but in fact you should go a step further. [Driver developers recommend](https://developer.nvidia.com/vulkan-memory-management) that you also store multiple buffers, like the vertex and index buffer, into a single [VkBuffer](https://www.khronos.org/registry/vulkan/specs/1.0/man/html/VkBuffer.html) and use offsets in commands like [vkCmdBindVertexBuffers](https://www.khronos.org/registry/vulkan/specs/1.0/man/html/vkCmdBindVertexBuffers.html). The advantage is that your data is more cache friendly in that case, because it's closer together. It is even possible to reuse the same chunk of memory for multiple resources if they are not used during the same render operations, provided that their data is refreshed, of course. This is known as *aliasing* and some Vulkan functions have explicit flags to specify that you want to do this.

在前一个章节的基础上修改为四个顶点（注意背面剔除）

利用inputAssembly.topology = VK\_PRIMITIVE\_TOPOLOGY\_TRIANGLE\_STRIP;的效果

