**Draw Managment**

**DrawIndexedObject** est l’objet qui défini un ou plusieurs objet à dessiner avec la fonction vkDrawIndexed

struct DrawIndexedObject

{

VkDescriptorSet\* descriptorSets; //explicit

VkPipelineLayout\* pipelineLayout;

VkPipeline\* pipeline;

VkBuffer\* vertexBuffer; //The buffer where the data are located

VkDeviceSize\* pVertexOffset; //The offset inside the buffer where the data are located

VkBuffer\* indexBuffer; //The buffer where the indices are located (it CAN be the same as the vertexBuffer)

VkDeviceSize indexOffset; //The offset inside the buffer where the indices are located

uint32\_t indexCount; //is the number of vertices to draw.

uint32\_t instanceCount; //is the number of instances to draw.

uint32\_t firstIndex; //is the base index within the index buffer.

uint32\_t vertexOffset; //is the value added to the vertex index before indexing into the vertex buffer.

uint32\_t firstInstance; //It's the instance ID of the first instance to draw.

} DrawIndexedObject;

Maintenant nous devons « build » les commandBuffer de dessin (drawCommandBuffer) avec comme base un DrawIndexedObject

std::vector<DrawIndexedObject> m\_drawObjects;

void buildCommandBuffer()

{

for (size\_t i = 0; i < m\_drawObjects.size(); i++) {

vkCmdBindDescriptorSets(cmdBuffer, VK\_PIPELINE\_BIND\_POINT\_GRAPHICS,

&m\_drawObjects[i].pipelineLayout, 0, 1, m\_drawObjects[i].descriptorSets, 0, nullptr);

vkCmdBindPipeline(cmdBuffer, VK\_PIPELINE\_BIND\_POINT\_GRAPHICS, &m\_drawObjects[i].pipeline);

vkCmdBindVertexBuffers(cmdBuffer, ?, 1, m\_drawObjects[i].vertexBuffer, m\_drawObjects[i].pVertexOffset);

vkCmdBindIndexBuffer(cmdBuffer, m\_drawObjects[i].indexBuffer, m\_drawObjects[i].indexOffset, VK\_INDEX\_TYPE\_UINT32);

vkCmdDrawIndexed(cmdBuffer, m\_drawObjects[i].indexCount, m\_drawObjects[i].instanceCount, m\_drawObjects[i].firstIndex, m\_drawObjects[i].vertexOffset, m\_drawObjects[i].firstInstance);

}

}