Summarizing Notes from Meeting with Martin Kraus.

The impression of Vulkan is that it gives the programmer more flexibility at the cost of programmability. It makes sense to build an API on top of Vulkan.

Most of his lectures focus on shader programming in Unity, however some students learn a bit of OpenGL.

He has worked with graphics programming for many years, while his master's degree is in physics, his ph.d is within the field of graphics.

Currently works with graphics for virtual reality.

Comments on basic example

- Basic Example A
    - Assumes that a texture refers to a texture resource ID rather than the resource itself.
    - Assumes that render pass is associated with shaders.
    - Thinks that the window surface may be a render target. Not sure why three frambuffers is used in the example.
    - Thinks that texture sampling should indicate wheteher it is used for 2D or 3D sampling.
    - *SetRenderPasses* required some explanation.
    - Not sure when commands are executed. Are they sequential?
    - Would like the binding between vertex input and shader to be more explicit.
    - What is the difference between primary and secondary command buffers?
- Basic Example B
    - It is more natural to compile shaders with method *compileFromFile.*
    - Thinks the type *VertexShaderByteCode* should be renamed to *VertexShader*.
    - Likes the idea of a color class.

Comments on shadow mapping example

- Shadow mapping example A
    - A 2D texture should be referred to as a texture2D.
    - Handling of resources should be more explicit. Difficult to follow
    - Not sure about how off-screen textures are handled.
- Shadow mapping example B
    - What is the difference between a *Renderer* and a *TextureRenderer*?
    - Writing dependencies between passes is surprising.

Comments on Multi Threading example

- Doesn’t have experience with multi-threaded programming on the CPU
- Not sure how work is passed out to the different threads
- Did not expect a performance boost from multi threaded command construction. More used to GPU-bound applications.

Impression of the API

It looks interesting and it would be beneficial to have something people can learn to use on top of Vulkan. However how does our API compete with OpenGL?