How to learn Vulkan

https://www.jeremyong.com/c++/vulkan/graphics/rendering/2018/03/26/how-to-learn-vulkan/

Lina Liu Non. 14 2022

Why you learning Vulkan?

1. Vulkan performs better
2. Vulkan provide a more viable cross-platform solution(MoltenSDK is open sourced)
3. Vulkan is of course shiny things
4. Vulkan can be multithreaded programming
5. Vulkan is less about graphics, and more about GPU drivers at its core.
    1. Vulkan not provide how to cascaded shadow map works
    2. Vulkan is not screen-space-relections
    3. Vulkan is not how indirect lighting is done.
6. Vulkan is very broad and deep but it is logical
7. Learning Vulkan means jumping into a very large codebase.
8. Vulkan is best by using C++ instead of interpreted languages, why?
    1. The overhead of the extra function calls will add up and potentially offset the performance gains.

How to learn Vulkan?

How to adjust your mentality to best maximize your chance for success?

1. Don’t discouraged if getting to this draw takes 5 to 10 times longer than you’re used to.
    1. To bloster performances, as everything is opt-in
2. You need to constantly backtrack and review.
    1. To wonder why something is necessary?
    2. To wonder how something was done?
    3. Graphics pipeline is extremely deep, it’s easy to lose the big picture once you get stuck in the weeds.
        1. You need to proactively pause and remind yourself what you have done and accomplished in order to get to where you currently are.
3. Many existing frameworks and engines proved existing functionality to Vulkan, make abstraction choices to be compatible with Vulkan.
4. Refer to the spec early and often.
    1. It in SDK docs folder.
    2. Useful information in the Vulkan Spec to be quite high and well worth the time invested.