Cross-Platform Vulkan with GLFW
http://web.engr.oregonstate.edu/~royl/vulkan-glfw.odp

Lance Roy

Vulkan is Headless

- Vulkan API provides no way of drawing to the screen
- Need to use extensions
  - VK_KHR_surface
  - VK_KHR_swapchain
- Need to get a surface
  - This requires platform dependent extension.
- Creating a window is also platform dependent.

XCB

- VK_KHR_surface, VK_KHR_xcb_surface
- Need to give Vulkan a window
- Similar for Xlib

typedef struct VkXcbSurfaceCreateInfoKHR {
  VkStructureType             sType;
  const void*                 pNext;
  VkXcbSurfaceCreateFlagsKHR  flags;
  xcb_connection_t*           connection;
  xcb_window_t                window;
} VkXcbSurfaceCreateInfoKHR;

VkResult vkCreateXcbSurfaceKHR(
  VkInstance                                  instance,
  const VkXcbSurfaceCreateInfoKHR*            pCreateInfo,
  const VkAllocationCallbacks*                pAllocator,
  VkSurfaceKHR*                               pSurface);

Win32

- VK_KHR_surface, VK_KHR_win32_surface
- Same idea, but need a Win32 window.

typedef struct VkWin32SurfaceCreateInfoKHR {
  VkStructureType                 sType;
  const void*                     pNext;
  VkWin32SurfaceCreateFlagsKHR    flags;
  HINSTANCE                       hinstance;
  HWND                            hwnd;
} VkWin32SurfaceCreateInfoKHR;

VkResult vkCreateWin32SurfaceKHR(
  VkInstance                                  instance,
  const VkWin32SurfaceCreateInfoKHR*          pCreateInfo,
  const VkAllocationCallbacks*                pAllocator,
  VkSurfaceKHR*                               pSurface);

Window Creation Library

- We need to do the same thing on all platforms
- We can do it in the same way on all platforms
- Seems like a library would be useful

GLFW

- Cross-platform
  - Windows
  - Linux
    - X11
    - Wayland
    - Mir
  - Mac (through MoltenVK)
- Cross-API
  - Vulkan
  - OpenGL
GLFW: Initialization

- Initialize GLFW
- Create the window.

```cpp
glfwInit();
MainWindow = glfwCreateWindow( Width, Height, "Vulkan Sample", NULL, NULL );
```

(from sample.cpp)

GLFW: Vulkan

- Get the required extensions
  - Pass them into `vkCreateInstance`
- Create the Vulkan surface.
  - Can then be used to create the Swapchain.

```cpp
int glfwVulkanSupported();
const char** glfwGetRequiredInstanceExtensions(uint32_t* count);
VkResult glfwCreateWindowSurface(VkInstance instance, GLFWwindow *window, const VkAllocationCallbacks *allocator, VkSurfaceKHR *surface);
```

Mac

- Apple wants developers to use their own low-level graphics API
- No Official Vulkan support
- Third party, commercial compatibility layer called MoltenVK
  - GLFW support

Metal

- Metal

Mobile

- Vulkan works on Android
  - Not supported by GLFW (yet).

Making sample.cpp work on Linux

- http://web.engr.oregonstate.edu/~royl/sample-vulkan-linux.patch
- I will quickly run through the changes
- Compilation Command:

```
g++ -O2 sample.cpp -DPERMISSIVE --std=c++11 -lvulkan -lglfw -o sample
```