GLFW

Mike Bailey
mjb@cs.oregonstate.edu

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License

http://www.glfw.org/

GLFW is an Open Source, multi-platform library for OpenGL, OpenCL, EGL and Vulkan development (and development of applications) focused on creating window, contexts and surface, receiving event and rendering.

GLFW is written in C and has native support for Windows, macOS, Linux and many Unix-like environments on the X Window System, such as Linux and FreeBSD.

GLFW is license with the zlib/png license.

- Gives you a window and OpenGL context with just two function calls
- Support for OpenGL, OpenCL, Vulkan and related options, flags and extensions.
- Support for multiple windows, multiple monitors, high DPI and gamma ramps.
- Support for keyboard, mouse, gesture, time and window events input, via polling or callbacks.
- Comes with guides, tutorials, reference documentation, examples and build programs.
- Open Source with an OSG certified license allowing commercial use.
- Access to native objects and compile-time options for platform specific features.
- Community-maintained bindings for many different languages.

No library can be perfect for everyone. If GLFW doesn’t meet you’re looking for, there are alternatives.
Setting Up GLFW

```c
void InitGLFW( )
{
    glfwInit( );
    glfwWindowHint( GLFW_CLIENT_API, GLFW_NO_API );
    glfwWindowHint( GLFW_RESIZABLE, GLFW_FALSE );
    MainWindow = glfwCreateWindow( Width, Height, "Vulkan Sample", NULL, NULL );
    VkResult result = glfwCreateWindowSurface( Instance, MainWindow, NULL, &Surface );
    glfwSetErrorCallback( GLFWErrorCallback );
    glfwSetKeyCallback( MainWindow, GLFWKeyboard );
    glfwSetCursorPosCallback( MainWindow, GLFWMouseMotion );
    glfwSetMouseButtonCallback( MainWindow, GLFWMouseButton );
}
```

GLFW Keyboard Callback

```c
void GLFWKeyboard( GLFWwindow * window, int key, int scancode, int action, int mods )
{
    if( action == GLFW_PRESS )
    {
        switch( key )
        {
            //case GLFW_KEY_M:
            case 'm':
                case 'M':
                Mode++;
                if( Mode >= 2 )
                    Mode = 0;
                break;

            default:
                fprintf( FpDebug, "Unknown key hit: 0x%04x = "]key", key );
                fflush(FpDebug);
        }
    }
}
```
while( glfwWindowShouldClose( MainWindow ) == 0 )
{
    glfwPollEvents( );
    Time = glfwGetTime(); // elapsed time, in double-precision seconds
    UpdateScene( );
    RenderScene( );
}

vkQueueWaitIdle( Queue );
vkDeviceWaitIdle( LogicalDevice );
DestroyAllVulkan( );
glfwDestroyWindow( MainWindow );
glfwTerminate( );