Pointers, Arrays, and References

In C++, arrays are used to store a fixed-size sequence of elements of the same data type. To create an array in C++, you need to follow certain requirements:

Alternatives to Pointer:

Looking at Pointer heavy code is difficult to determine intent of the programmer. Unless you are writing libraries you should almost never have to worry about manual allocation and deallocation. Memory for objects should be allocated in a constructor, and then deallocated in the destructor. Use containers such as `std::vector` (for resizable arrays) and `std::array` (for fixed size arrays) instead of the naive array type. They are essentially zero-overhead, take care of all allocation, reallocation (in the case of vector) and deallocation, and provide a fairly comprehensive interface.

*Consider using the Standard Library before using **pointers*** for general Software Engineering. In modern C++ (11 and later)

- *To hold a collection of values,* consider a container, such as **vector, set, map, unordered_map,** or **array**
- *To hold a string of characters,* consider **String**
- *To point to an object you own* (i.e., must delete) use **unique_ptr,** or **shared_ptr**
- *To point to a continguous sequence of elements that you dont own*, use **span**
- *To systematically avoid dereferencing a null pointer*, use **not_null**

Pointers are not use in C-style Shader languages like GLSL/HLSL, but is critical to understand in Vulkan, and to lesser degree OpenGL.

Revision #4
Created 21 September 2024 20:56:23 by victor
Updated 25 September 2024 21:50:16 by victor