The goal for this library is to provide a modular set of subsystems which may be combined as a cohesive simulation engine. As I see it, games are a subset of simulations, so this engine would not be limited to games, though it would be capable of creating games.

I would like for the library to allow a large degree of customization (by programmers; in other words, to be an extensible API) while being clean and easy to work with. It should also provide a lot of basic functionality built-in. For example, there should be at least one basic view component fully implemented which provides the basic functionality you’d expect from a window and 2-d, blit-based renderer. At the same time, it would allow for users to extend basic functionality by creating their own view however they like based on the framework laid out in the engine.

Ultimately, I would like for the library to get to the point where running a simulation is as simple as:

#include”avl library.h”

int WinMain(blah blah)

{

avl::Simulation application(“simulation properties.avl”);

return application.Run();

}

Or something along those lines.

I would also like to eventually create a toolkit for creating basic resources, scripts, worlds, etc. This library is for my use, and I want to it to be easy to use and extend for myself for years to come. Each subsystem is independent (except that they all depend on utility), and it’s possible to use components individually. This will allow me to use the functionality that I need when I need it.