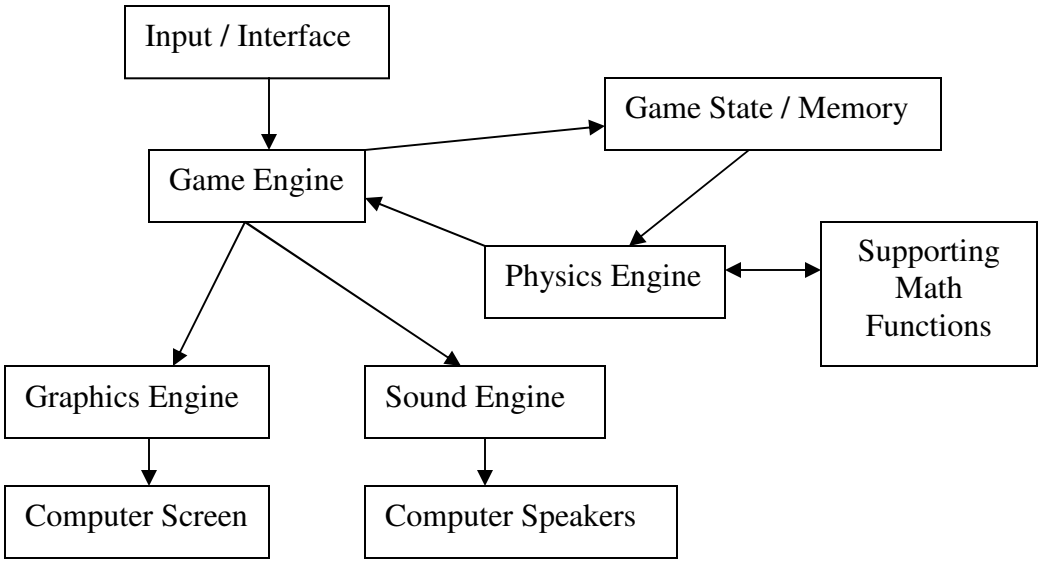
Mark Walker
Senior Project Presentation
October 17, 2005

Program Architecture



The overall layout of how the program functions must be designed beforehand in order to avoid complications later on. Kind of like baking a cake, you need to have a recipe before you get started so you know what ingredients you need, how to mix them, how long to cook it for, etc., or else you could end up mixing several batters before you get it right.

The physics engine is one of many components that makes the game run. It is responsible for all physical world simulations (this could be though of as an automatic physics problem solver). Most physics engines aren't designed for any one game – they can be implemented into many other similar games as well. In other words, the game developers should not have to worry about changing the physics engine. In fact, usually they are not even allowed to if they have a license to a 'closed source' engine, and are only able to utilize it through built-in functions.

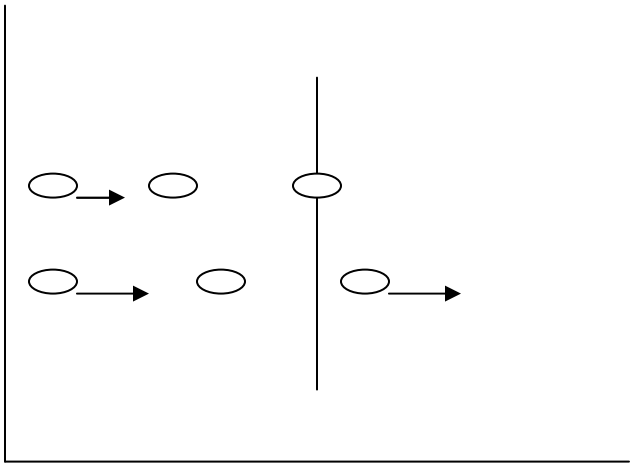
Basically program architecture just determines "who does what," so that all the components run together well.

Supporting Classes
A few supporting classes involving math will be needed. A vector class will be needed for sure, and probably some sort of coordinate system will be needed. This depends a lot on the program architecture.

Solid Body Masses and Collision Detection
This will probably be the hardest part of the project, which is why I bought an entire book on it. Challenges will be:

- How to represent objects geometrically (squares, circles, polygons)

- How to detect when two or more objects collide *efficiently*
    - Becomes a very slow process [$O(n^2)$] with more objects
- When more than two objects collide simultaneously
- How the objects react when they collide
    - Non-ideal when friction and air resistance are introduced
- The bullet through the wall problem and other special cases



Other Challenges

- Overall efficiency
- How easy it is to implement and use
- How accurate is it
    - Designed primarily for games
    - Certain properties of computers render them prone to seemingly small mathematical glitches that can end up having a huge effect in short time
    - May be hard to determine if the physics itself has been coded correctly just by watching it
- Surprises!

Purpose of the Project

Besides being a very useful experience for myself as both a physics student and programmer, this engine could be used by other people as well. From the web searching I have done, I have not found a 2-dimensional physics engine in Java quite as comprehensive as the one I have proposed. Most of them are educational tools that have very specific purposes, but could not be used for anything else. At the end I could post my work on the web for others to both learn from and use in their own programs. If I have time at the end, it may be worthwhile to document the process and the outcome for other people who may be interested in developing similar programs.