CS 230

Game Implementation Techniques

1 Game Engine Design

1.1 Overview

- A game engine is divided into 2 components: System component and Game Logic component.
- These two components should be totally separated. One deals strictly with the hardware while the other deals with game logic.
- Each subcomponent should belong entirely to either one of these two components. If a subcomponent turns out to be contributing to both components, it should be divided to 2 or more several subcomponents, where each of them satisfies the aforementioned rule.

1.2 System components: Introduction

- The system component communicates with the hardware.
- It isolates the game logic component from directly communicating with the hardware.
- This isolation makes it easier to port the game to another platform.
- The system component is broken into several subcomponents: Memory manager, graphics manager, game state manager, audio manager, frame rate controller...

1.3 Game logic components: Introduction

- The game logic component is where the game code resides.
- It does not communicate directly with the hardware.
- It does not even know what the underlying hardware is.
- When the game code needs to communicate with the hardware, it uses system components functionalities in order to do so.
- It also contains the physics/collision engine, object manager, environment manager, messaging system, camera system...

- **Managing system components**
  Any game code (or application in general) has to initialize some system components before actually going into the game code and the game loop. These system components usually set up some necessary hardware related functionalities which will be later used within the game. Example:

  - Setting up a input device
  - Setting up video device
  - Setting up an audio device
  - Allocating video buffers
  - Deciding if some parts of the pipeline should be done using the hardware or software.

  This kind of system component initialization should be done just once before entering the game loop, and if any component fails to initialize properly, the application or the game usually quits with the appropriate error, since the application won't be able to run.
  If all system components are initialized properly, we initialize few arguments like the frame rate controller and the previous/current/next game state and the game loop is started. Note that the previous code should never be reached again.

  Upon exiting the game loop, all the devices that were allocated should be released before exiting the application. For example, if a video device was created, it should be released in order to free all its allocated resources on the video card. Also, if an input device was allocated during the initialization stage, it should be released upon exiting the application. This would be the final code part of the game or the application, and care must be taken to make sure every allocated resource is released.

