Game Engine Architecture

CMSC 20900 - Computers for Learning

Some slides courtesy of Mark Floryan (University of Virginia)
Outline

- Event-Driven Programming
- Game Loop:
  - Update
  - Draw
Event-Driven Programming

- A programming paradigm where the flow of the program is determined by events such as user actions, sensor outputs, or messages from other programs/threads
Event-Driven Programming

- Game Loop
  - User Input
  - Physics/Movement
  - Anything happening in real time
    *Will cover this today

- Events in Game
  - Loading/Initializing game levels
  - Completing game levels
  - Updating User Interface, etc
    *Will cover this later in the quarter
Game Loop

60 times/s = 16.7 ms

- Update
  - Receive player input
  - Process player actions
  - Process NPC actions
  - Post-process (e.g. physics)

- Draw
  - Cull non-visible objects
  - Transform visible objects
  - Draw to backing buffer
  - Display backing buffer
Update Loop Step 1: Receive Player Input

- Player input is one set of variables that can affect game state
- Input is typically collected during the game loop
  - What is the state of the controller/keyboard?
  - If no change, do nothing
- We can only read input once per game loop cycle.
- But frame rate is short and most events are longer than one frame
Update Loop Step 2: Process Player Actions

- Alter the game state based on your input
- But you don’t want to directly change the state of the Object with user input
  - Do this by calling a method (data is private)
Update Loop Step 3: Process Non-Player Characters

- A NPC is anything that has volition in the world that is not your character (i.e. ghosts in Pacman)
- Work on the idea of Sense-Think-Act:
  - Sense the state of the world around it
  - Think about what action to perform
  - Act in the world
Update Loop Step 4: World Processing

- Physics!
- Collisions!
- *We will do this in future labs :)*
Draw Loop

- Needs to be fast!
  - We want to do as little computation as possible
  - Draw ONLY what needs to be on the screen
- Keep the drawing and the state modification separate!
- Some of this will be handled by Java’s graphics libraries
Pre-Lab 1: Intro to Java & Game Engine

- Learn basic Java syntax and concepts
- Look at Game Engine scaffold:
  - You will see a basic implementation of the update and draw loop
Architecture Big Picture

[Diagram showing game engine components: Input Devices, Player, GUI, Rendering Engine, Audio Engine, Compiler, Data Management Layer, Character Scripts, Character Data, UI Elements, Models and Textures, Sounds.]