Game Engine Flow

- Load program
- Initialize variables
- Load mission/level information
- Place objects/NPCs into world
- Schedule events
- Start clock
- Spawn player
- Handle events
  - Generated by player(s), NPCs, or timers
Multiplayer: Server

- Start server
  - Like previous slide
  - Events include clients joining
- Spawn player
- Receive updates from clients
- Update global state
  - Maintain the world state
- Disseminate state changes
  - To clients
  - To other servers

Multiplayer: Client

- Load client code
- Search for a server
  - Choose wisely!
- Establish connection
- Receive current game state
- Render game to user
- Receive
  - Input from user
  - Updates from server
Game Engines

- Scene graph
  - Representation of the world
  - Includes characters
- Timing is very important
  - Events
    - Time-based
    - Multi-player
  - Synchronization
- Database of objects
- Networking
  - Between Server and clients
  - Between Servers

Game Graphics

- Different from other media
  - Need to process and display @ 30 fps
  - Dynamic scenes
- Graphics Processing Units (GPUs) are now programmable
  - Need to understand how to program for them
  - nVidia's cg programming language
  - OpenGL 2.0 extensions
  - Stream-processing model
  - Data must be packed into textures
  - Limited control support
    - Loops, stack data structures
- Good jobs here!
Physics

- Need to consider how fast you can compute
  - Scalable in the number of objects?
  - Scalable in the types of objects?
    - Cloth?
    - Hair?
    - Water?
- Three main types of objects
  - Point masses
  - Rigid bodies
  - Soft bodies
- Life is a combination of physics and freewill
  - How do we balance these?

Server Details

- Server performs multiple tasks concurrently
  - Each WORKER is a separate thread
  - How do they coordinate efforts?

  - Initialize game
    - Load assets
    - Load map
    - Load add-ons (mods)
    - Initialize NPCs
    - Start WORKER threads
    - Start clock

  - WORKER
    - Update NPC state
      - Time
      - Behavior

  - WORKER
    - Client join/leave process

  - WORKER
    - Transmit updates to clients
      - Object state
      - Player state
      - Does this scale?

  - WORKER
    - Listen for client updates
      - One port per client?
Server Coordination

- Each worker has tables of interest
  - Workers sleep until table data changes
  - Database dispatcher monitors tables, wakes workers

---

Even More Server Details

- For this to work, you need
  - Threads
  - Inter-process/thread Communication
    - Sockets
    - Shared memory
  - Some way of doing timing
    - Callback
    - Interrupt handler
  - An efficient data store

- In order to do it well, you also need
  - Thorough understanding of systems programming
  - A very good design, and lots of it!
  - You should have seen this in CS-3013: OS, and CS-2303: Systems Programming Concepts
Client/Server Approach

- Requires messages to be passed
  - Network could be bottleneck
  - Server could be bottleneck

- Lag is bad
  - Example: the player you shoot at is "magically" not there anymore by the time the projectile gets to him

- Inconsistent state is bad
  - Who grabbed that object first?

Client/Server Programming

- Make it easy on the programmer
  - Hide the fact that things are being sent to server

- Make "surrogates" for server objects
  - Underlying system does actual communication

- How can we make a system really scalable to 1000s of users?
  - How is this done in gaming systems?
Graphical User Interface

- Provides access to
  - Game menus (*e.g.*, save, load, boss)
  - Player status (*e.g.*, health, current speed)
  - Maps
    - Current play location
    - Location of "persons of interest"
    - Location of "goals"
  - Non-Player Character (NPC) dialog
  - Player-to-player chat

C4 Engine Structure

- Layered structure
  - Base Services
  - System Managers
  - Large-Scale Architecture
  - Plugin Modules

C4 Base Services

- File Manager
- Memory Manager
- Time Manager
- Resource Manager
- Math Library
- Utility Library
- System Utilities

C4 System Managers

- Sound Manager
- Movie Manager (animated textures)
- Display manager
- Graphics Manager
- Input Manager
- Network Manager
C4 Large-Scale Architecture

- Interface Manager
- Message Manager
- Effect Manager (fluid, cloth, particles)
- Scene Graph
- Animation System
- Controller System
- Tool Manager
- World Manager
- Lighting & Shadowing

C4 Plugin Modules

- Import Tools (Collada, TGA files)
- World Editor
- Application Module
- Media players
  - Model viewer
  - Texture viewer
  - Font generator
  - Sound player
  - Movie player