CS425 GAME PROGRAMMING

Level Loading and Management

Read GEA Chapters 8.1 (Rendering loop) & 8.2 (Game Loop) & 7 (Resources & File system)

Announcements

- PA #00 is due tonight
- PA #01 will be out in a day or two

Today

- Review of Game Engine/Event-Based System
- Game Loops
- Loops and events in SDL
- Level loading

Review: Brute-force manual start-up and shut-down method

```
int main(int argc, const char* argv)
{
    // Start up engine systems in the correct order.
    gMemoryManager.startUp();
    gFileSystemManager.startUp();
    gVideoManager.startUp();
    gTextureManager.startUp();
    gRenderManager.startUp();
    gAnimationManager.startUp();
    gPhysicsManager.startUp();
    // ...

    // Run the game.
    gSimulationManager.run();

    // Shut everything down, in reverse order.
    // ...
    gPhysicsManager.shutDown();
    gAnimationManager.shutDown();
    gRenderManager.shutDown();
    gFileSystemManager.shutDown();
    gMemoryManager.shutDown();

    return 0;
}
```

The Game Loop

The Rendering/Game Loop

```
while (!quit)
{
    // Update the camera transform based on interactive
    // inputs or by following a predefined path.
    updateCamera();

    // Update positions, orientations and any other
    // relevant visual state of any dynamic elements
    // in the scene.
    updateSceneElements();

    // Render a still frame into an off-screen frame
    // buffer known as the "back buffer".
    renderScene();

    // Swap the back buffer with the front buffer, making
    // the most recently rendered image visible
    // on-screen. (Or, in windowed mode, copy (blit) the
    // back buffer's contents to the front buffer.
    swapBuffers();
}
```

Message/event-based System

```
while (true)
{
    // Service any and all pending Windows messages.
    MSG msg;

    while (PeekMessage(&msg, nullptr, 0, 0) > 0)
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    // No more Windows messages to process -- run one
    // iteration of our "real" game loop.
    RunOneIterationOfGameLoop();
}
```

Framework-based Loop

```
while (true)
{
    for (each frameListener)
    {
        frameListener.frameStarted();
    }

    renderCurrentScene();

    for (each frameListener)
    {
        frameListener.frameEnded();
    }

    finalizeSceneAndSwapBuffers();
}
```

Framework-based Loop (Cont.)

```
class GameFrameListener : public Ogre::FrameListener
{
public:
    virtual void frameStarted(const FrameEvent& event)
    {
        // Do things that must happen before the 3D scene
        // is rendered (i.e., service all game engine
        // subsystems).
        pollJoypad(event);
        updatePlayerControls(event);
        updateDynamicsSimulation(event);
        resolveCollisions(event);
        updateCamera(event);

        // etc.
    }

    virtual void frameEnded(const FrameEvent& event)
    {
        // Do things that must happen after the 3D scene
        // has been rendered.
        drawHud(event);

        // etc.
    }
};
```

Pong: Game Loop

- How do you design a game loop for Pong

Pong: Game Loop

- How do you design a game loop for Pong

Pong: Game Loop (Chapter 8.2)

```
void main() // Pong
{
    initGame();

    while (true) // game loop
    {
        readHumanInterfaceDevices();

        if (quitButtonPressed())
        {
            break; // exit the game loop
        }

        movePaddles();

        moveBall();

        collideAndBounceBall();

        if (ballImpactedSide(LEFT_PLAYER))
        {
            incremenentScore(RIGHT_PLAYER);
            resetBall();
        }
        else if (ballImpactedSide(RIGHT_PLAYER))
        {
            incrementScore(LEFT_PLAYER);
            resetBall();
        }

        renderPlayfield();
    }
}
```

Design a Game Loop

SDL2

- It has in fact multiple libraries, make sure to use the necessary library
  - SDL2
    - Core SDL2 functions
  - SDL2_image
    - For loading png and other image formats
  - SDL2_ttf
    - For loading true type font
  - SDL2_mixer
    - For sound and music
  - SDL2_network
    - For basic network communication
  - …

SDL2

- Basic Structure of SDL2 code

```
#include <SDL.h>
int main(int argc, char *argv[])
{
        SDL_Init( … );
        SDL_CreateWindow( … );

        //Load game resources

        //the Game Loop
        while( true )
        {
          SDL_Event e;
          while( SDL_PollEvent( &e ) != 0 )
          {
                  //handle event e
          }
        }
        SDL_DestroyWindow( … );
}
```

SDL2 events

- SDL_Event
  - SDL2 defines many events for IO, GUI and timer
  - Most important data: event.type
- SDL_PollEvent(SDL_Event * event)
- SDL_PushEvent(SDL_Event * event)
  - You can also define your own event

```
SDL_Event user_event;
user_event.type = SDL_USEREVENT;
user_event.user.code = 2;
user_event.user.data1 = NULL;
user_event.user.data2 = NULL;
SDL_PushEvent(&user_event);
```

SDL2 Drawing

```
#include <SDL.h>
int main(int argc, char *argv[])
{
        SDL_Init( … );
        SDL_CreateWindow( … );
        SDL_Renderer * gRenderer = SDL_CreateRenderer( … )
        //Load game resources
        //the Game Loop
        while( true )
        {
          SDL_Event e;
          while( SDL_PollEvent( &e ) != 0 )
          {
                  //handle event e
          }
          SDL_SetRenderDrawColor( gRenderer, …);
          SDL_RenderClear( gRenderer );
          //Render current frame
          SDL_RenderPresent( gRenderer );
        }
        SDL_DestroyWindow( … );
}
```

Level Loading

- Why bother?
- What are we loading?
- What info do we need about what we’re loading?

Anatomy of a Game World

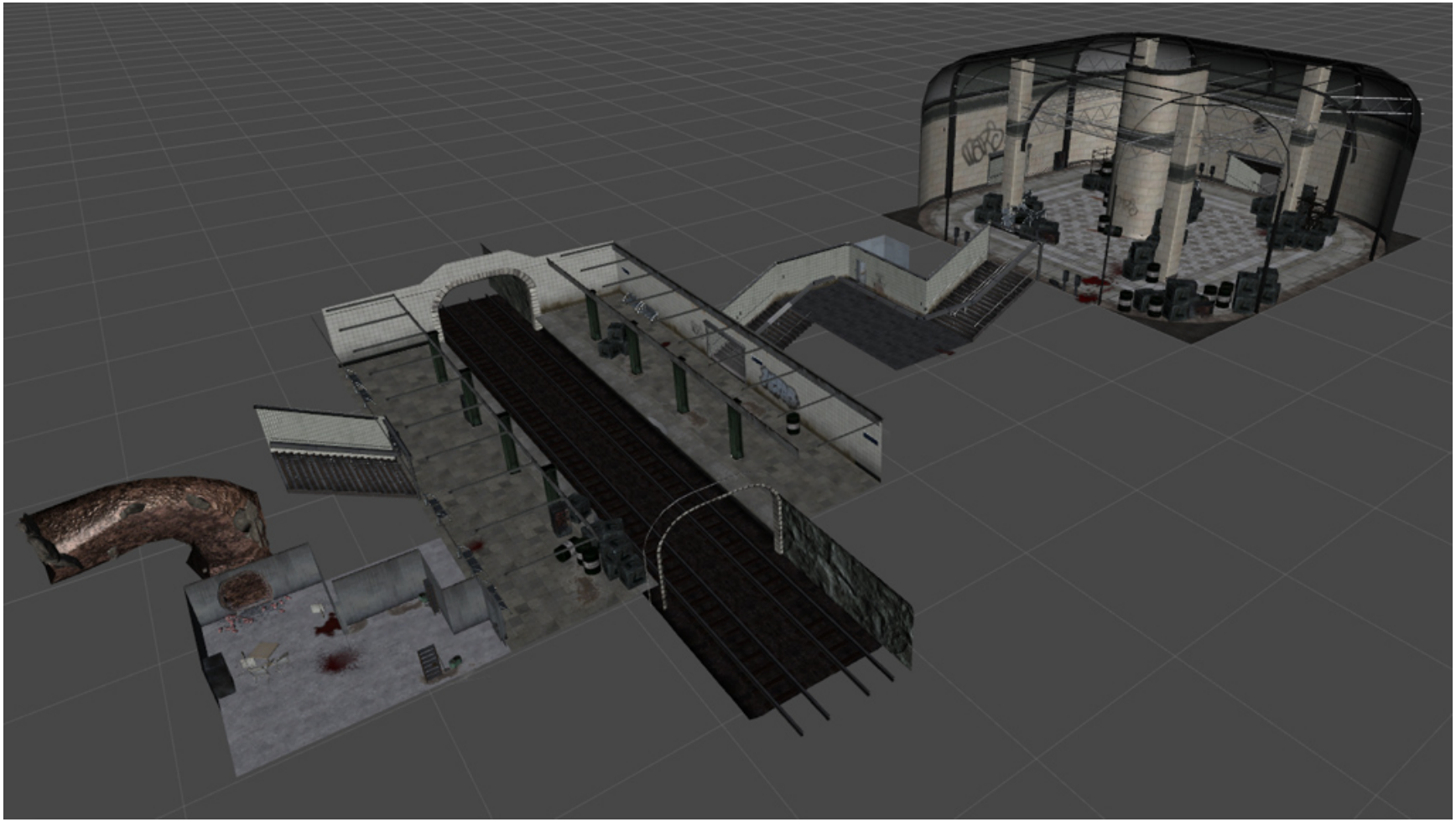
- World Chunks
- High-level Flow

- Static Background
- Dynamic Foreground

Skyrim

World of Warcraft

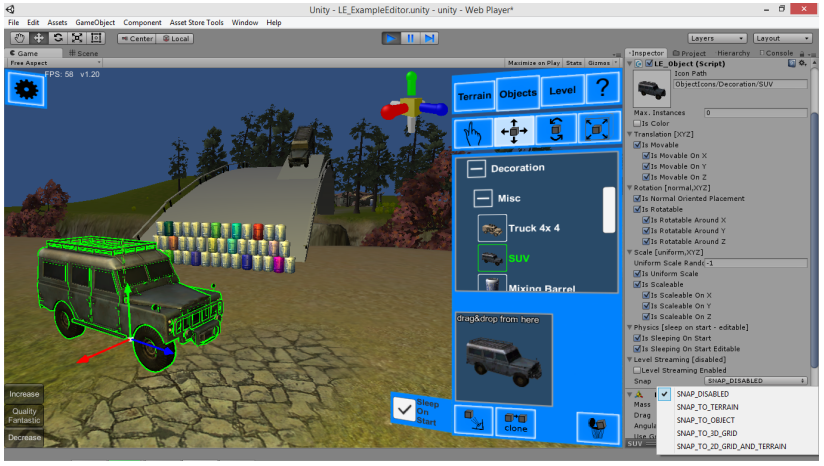
Level in Unity Editor

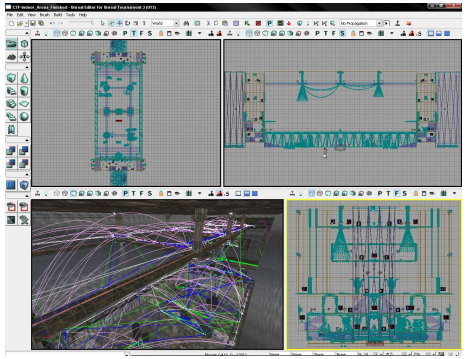


Even Scratch has it

Editor Features

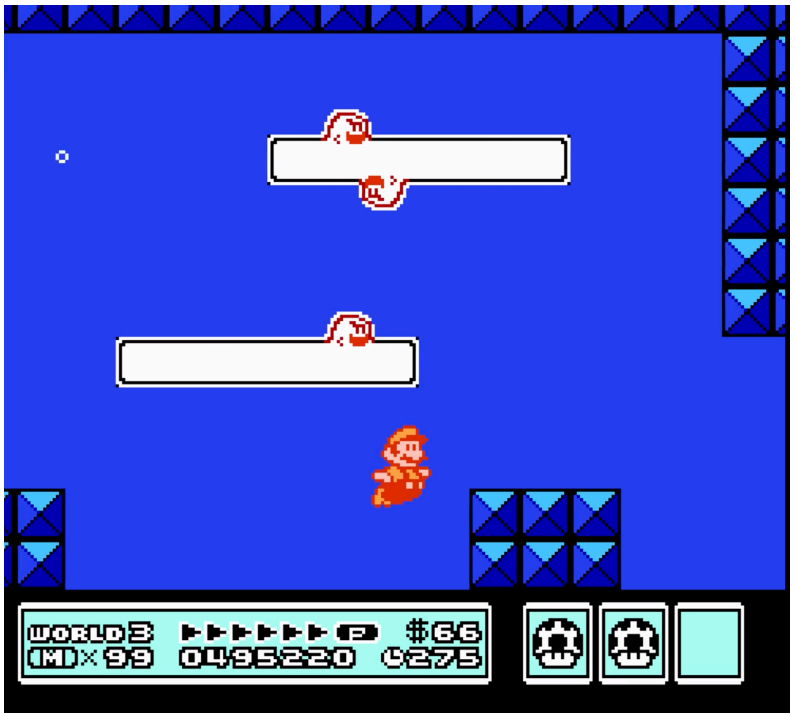
- Chunk creation and management
- Visualization
- Navigation
- Selection
- Layers
- Property grid (attributes)
- Object placement and alignment aids
- Special objects: lights, particle emitters, sounds, regions, splines, nav meshes, custom data, video





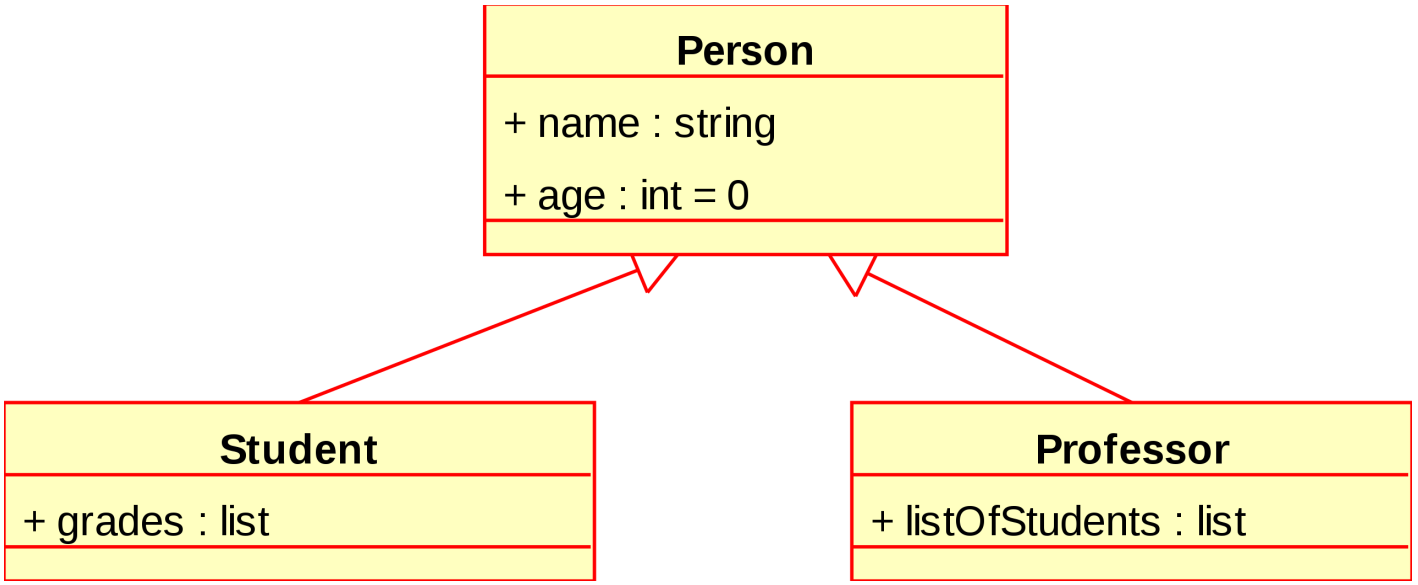
Design a Class Diagram

- Design a game engine that allows Nintendo to design this game
  - What are the events and where should the game loop go?
  - What are the assets and various resource managers?



Class Diagram

- A class diagram is a type of static structure diagram that describes the structure of a software system by showing the system's classes



Example

- Run a script
- Move React as a rigid body
- Emit Particles
- Play located audio
- Follow a path
- Animate

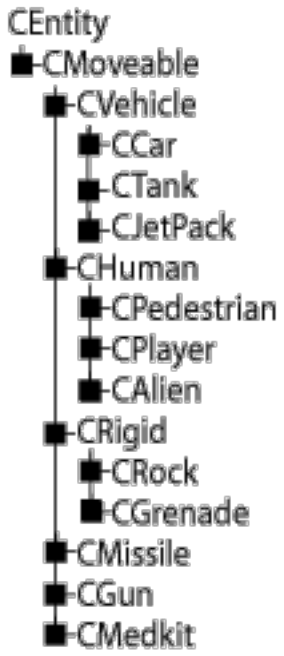


Figure 1 A typical game entity hierarchy.

- Evolve Your Hierarchy, Mick West

Your PA01

- Write a game engine that can load a level and move your character around
- A level would look like this:

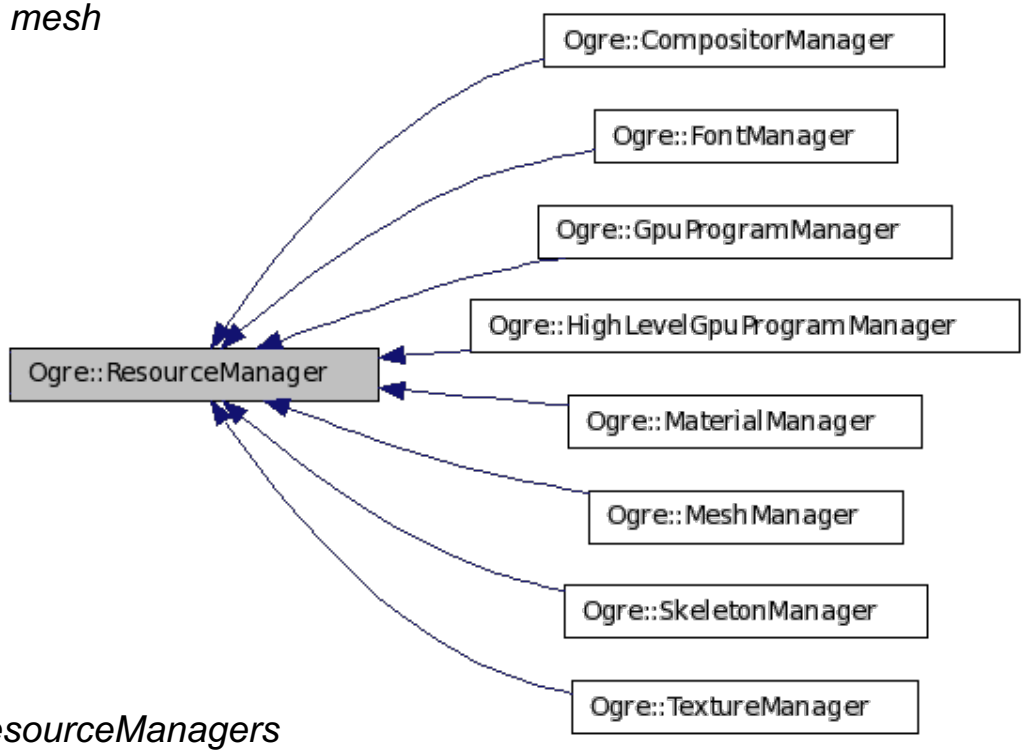
```
15 20
imgs/BackDrop.png

Objects
k knot.png 0.0 0.02
d dragon.png 0.0 0.04
t tudorhouse.png 0.0 0.01

Characters
s mario.png 15 3
b link.png 5.2 1

World
kooooosodoooook
oooooooooooooo
oooooboooooooo
oooooooooooooo
ooooooosoooooe
oooooooooooooo
oootooooooooo
ooooooooooosooo
owwwwwwooooooo
oooooowooosoooo
oboooowooooooo
oooooowooooooo
oooooowoootoooo
oooooooooooooo
oooooooooooooo
otooooooooooo
```

Asset management is no joke

- Loading and Unloading and Reloading
  - From where and to where?
  - Dependencies of assets (some assets require other assets to work together)
- Synchronous vs. Asynchronous Asset Manager
  - Loading everything upfront or when needed?
- Most modern game engines mange assets automatically
  - Unreal (**Asset Manager, Asset Tree**), Unity (**AssetBundle)**, etc.
  - Asset manager usually exists in both Editor and in the game

OGRE Resource Management

- Ogre needs to know where to search for "resource" files
  - Via configuration files (e.g., resources.cfg)
  - OGRE creates its resource manager in Root::Root
- When you say:
  - *Entity *e = mSceneMgr→createEntity("MyEntity", [MeshFilename]);*
  - *OGRE Resource manager finds the mesh*



See details in
http://wiki.ogre3d.org/Resources+and+ResourceManagers

Resources.cfg Example

```
# Resources required by the sample browser and most samples.
[Essential]
Zip=app/native/Media/packs/SdkTrays.zip
Zip=app/native/Media/packs/profiler.zip
FileSystem=app/native/Media/thumbnails

# Common sample resources needed by many of the samples.
# Rarely used resources should be separately loaded by the # samples which require them.
[Popular]
FileSystem=app/native/Media/fonts
FileSystem=app/native/Media/materials/programs
FileSystem=app/native/Media/materials/scripts
FileSystem=app/native/Media/models
FileSystem=app/native/Media/particle
…
Zip=app/native/Media/packs/cubemapsJS.zip
Zip=app/native/Media/packs/fresneldemo.zip
Zip=app/native/Media/packs/ogredance.zip
…
[General]
FileSystem=app/native/Media # Materials for visual tests

[Tests]
FileSystem=app/native/Media/../../Tests/Media
```

Runtime Object Model Architectures

- Object-centric vs. Property-centric
  - Class vs. ID and table of associated properties

- Problems with Deep, Wide Hierarchies
  - Understanding, Maintaining, and Modifying Classes
  - Inability to Describe Multidimensional Taxonomies
  - Multiple Inheritance: The Deadly Diamond
  - The Bubble-Up Effect (move functionality up the hierarchy so that more classes can share it, even if it is inappropriate)

Game Object Models

- Object oriented programming interface
- Adds functionality
- Tool-side vs. runtime
  - Designers vs. programmers
  - Different languages and tools
  - PIPELINE ISSUES!!!

Loaded Level

- Sample code