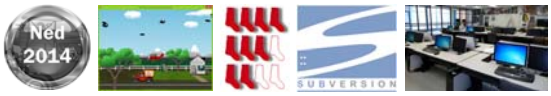


Syllabus

CSCE 4210/5250

Ian Parberry

General Information

- My name is *Ian Parberry*. If you wish to be formal, please call me *Dr. Parberry*.
- Contact: `http://larc.unt.edu/ian/`

OFFICE HOURS

Tue 2:30-3:30pm
Wed 4:00-5:00pm
Th 12:00-12:50pm
FTDP F209

Catalog Description

Introduction to game programming, including real-time, event-driven, and multimedia programming techniques. Graphics, sound and input programming. Students learn how to program a billboard game in 3D with constrained camera motion.

Learning Outcomes

1. Be familiar with Windows programming.
2. Be able to use Visual C++ 2013.
3. Be able to use the Microsoft Direct3D 11.2 SDK.
4. Be able to program a 3D billboard game.
5. Be able to work in a team with other programmers using Subversion.
6. Be able to code one or more aspects of a game, including graphics, sound, and gameplay.

Your Grade for This Class

- The grade for this class will be based on a project: create a simple computer game for Windows 8.1 using Visual Studio 2013 and DirectX 11.2.
- You will be using a revision control system.
- The game must run on the computers in the lab.
- There will be a game contest (not graded).
- You will present your game to me during Finals week, 30 minutes per group.

Prerequisites

- The listed prerequisite for this class is CSCE 2100: Computing Foundations I.
- You should be familiar with the basic data structures and formalisms used in computer science.

Expectations

You need to be or become:

- a strong C++ programmer.
- familiar with object-oriented design.
- comfortable with the following C++ concepts:
  - Derived classes
  - Virtual functions
  - Function and operator overloading
  - Memory allocation with **`new`** and **`delete`**
  - Default parameters
  - The keywords **`const`**, **`static`**, **`public`**, **`private`**, and **`protected`**

Expectations

This course is designed for students who intend to go into the game industry. I expect you to:

- spend a significant amount of time on this class starting from Day 1.
- behave like adults and not wait for me to spoon-feed you.
- read ahead in the lecture notes.
- experiment with my code.
- Google the subjects we cover in class and read more about them on the web.

Expectations

I expect you to:

- be excited and energized by the opportunity to learn new things.
- take responsibility for your education.
- be competitive.
- exceed expectations.

Cell Phone Policy

- Please put your cellphones on “stun”.
- However, having a cell phone ring in class is not the end of the world.

Class Structure

A typical class will look like this:

- A PowerPoint presentation. (Notes will be online.)
- Examining and experimenting with code live in class using Visual Studio. (Source code will be online.)
- Time in the lab for coding, debugging, group meetings, and further questions.

The Game Lab

- NTRP F204
- Open 40 hours per week (ish)
- Lab hours will be posted
- 22 Intel Core i7-3930K CPU @ 3.2GHz with 32GB of RAM (hex-core, hyperthreaded)
- Dual monitors, DirectX 11.2 capable video cards
- 64-bit Windows 8.1
- Visual Studio 2013

The Revision Control System

- You must use the Subversion server provided (more details later).
- I will be monitoring your progress during the semester.
- You must observe Best Practices when using the RCS (more later).
- I will be showing group progress in class towards the end of semester.

Homeworks

- There will be a few homeworks during the first half of the semester.
- These are Pass/Fail. If you don't do them, you will get an F for the class.
- They will be graded in class.
- They won't be difficult, so I expect you to overperform.

Important Dates

Week 15:

Game contest! Demo your game.

- Optional, will not affect grade
- Panel of industry judges

Finals Week:

Programmer interviews. Each group will get 30 minutes to show their game and answer instructor questions. A schedule will be drawn up towards the end of the semester.

The Milestone

Week ?

Some time around the middle of the semester I will call for your group to demonstrate your game to the class. This is called the *milestone*. Expect it.

My Grading Philosophy

I grade the executable, not the source code.

My grading scale

- A: it really knocks my socks off
- B: it's a pretty cool game
- C: it's an OK game
- D: it's not there, but at least you tried
- F: you really blew it off, didn't you?

Yes, this is not well-defined.
Welcome to the real world.

The Kind of Things I Look For

1. Completeness points: Is it finished?
2. Techno points: Does it contain things not covered in class?

Completeness points

- Does it run?
- Are there bugs?
- Is it actually a game?
- Is it fun?
- Does it engage the player?
- How far does it go beyond *Ned's Turkey Farm*?
- Is there sound?
- Did you use Subversion properly?
- Were you on time for the milestone?

Techno Points

Awarded for things not covered in class. Examples:

- Better graphics (lighting, bump mapping).
- Network play using WinSock.
- Level editor.
- Particle engine.
- Procedural content generation.

Why Windows and DirectX?

- You can learn graphics using Unix, gcc, and OpenGL in CSCE 4230: *Intro to Computer Graphics* and CSCE 5220: *Computer Graphics*.
- This is where you learn programming for Windows using Visual C++ and DirectX.
- We will be using DirectX 11.2. Windows 8.1, Visual Studio 2013, and fairly new graphics hardware is required for DirectX 11.2.
- The software is free from MSDNAA.
- If you can't afford the hardware, use the lab.

What Resources Can You Use?

- Almost anything goes.
- Don't recycle a complete game engine, but you may borrow and integrate code from any legal source.
- You will be graded on your contribution to the code.
- Be honest – attribute your work.
- Using code without acknowledging it to the instructor is cheating, and will be dealt with in accordance to the departmental cheating policy.

Incremental Development

- This class will be taught by showing the development of a simple game incrementally, i.e. step by step.
- A simple side-scroller: *Ned's Turkey Farm*
- The idea is not to teach you about a single game engine, but about game engine development using *Ned's Turkey Farm* as an example.

Demos for Ned 2014

- Demo 0: The Black Screen of Death
- Demo 1: Direct3D,
- Demo 2: Scripting and Debugging
- Demo 3: Sprites
- Demo 4: Animated Sprites & Scrolling Background
- Demo 5: Artificial Intelligence
- Demo 6: Scripted Level Design
- Demo 7: Sound


Ned Demo 0: The Black Screen of Death
Black Screen of Death: Hit Esc to Exit


Ned Demo 1: DirectX

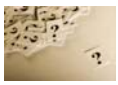

Ned Demo 3: The Sprite


Ned Demo 4: Animated Sprites and Scrolling Background


Ned Demo 5: Artificial Intelligence


Ned Demo 6: Scripted Level Design
NED'S

Why Teach This Way?



- I will not be teaching you everything there is to know about DirectX 11.2.
- You are capable of reading the documentation and the sample code.
- It is large and complex.
- Getting started is the hardest thing.
- I will show you how to get started in small, easy steps.

Ned Design Philosophy

- KISS: Keep It Simple, Stupid.
- *Ned* is an educational tool, not an all-purpose game engine.
- Teaching code ≠ production code.
- When introducing a concept, I try to keep it as simple as possible, but no simpler.
- This means that Ned is pretty bare-bones in some places.

Ned Design Philosophy

- This also means that *Ned* is a little lame.
- If you can see how to improve it, then you are ready for this class.
- If you can't see how to do better, maybe you should be taking another class.
- It's up to you to take it to the max.

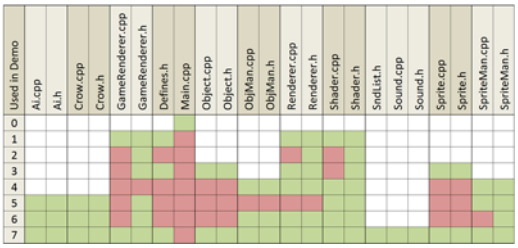
Tools

Used in Demo	ImageFileNameList.cpp	ImageFileNameList.h	Random.cpp	Random.h	Timer.cpp	Timer.h	tinyxml2.cpp	tinyxml2.h
0								
1								
2								
3								
4								
5								
6								
7								

Debug Tools

Used in Demo	Abort.cpp	Abort.h	Debug.cpp	Debug.h	IPMgr.cpp	IPMgr.h
0						
1						
2						
3						
4						
5						
6						
7						

Code



Lectures

1. Syllabus
2. Game Design
3. Visual Studio 2013
4. STL
5. 3D Math
6. Subversion
7. DirectX 11.2

0. Windows
1. DirectX 11.2
2. Scripting, Debugging
3. The Sprite
4. Animation
5. AI
6. Scripted Level Design
7. Sound

Cheating Policy

- The Department of Computer Science & Engineering cheating policy will be adhered to.
- Any student caught cheating *will receive a grade of F* for this course, and further disciplinary action will be taken.

DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING
CHEATING POLICY

- Cheating will be suspected if an assignment that calls for independent development and implementation of a program results in two or more solutions so similar that one can be converted to another by a mechanical transformation.
- Cheating will be suspected if a student who was to complete an assignment independently cannot explain both the intricacies of his or her solution and the techniques used to generate that solution.
- While it is not possible to give a complete definition of cheating, the following gives examples of clear instances of cheating and not cheating:

DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING
CHEATING POLICY

Cheating:

- Turning in someone else's work as your own. This includes, but is not limited to, another classmate, another textbook, a student in another class, or a student in a prior semester.
- Allowing someone to turn in your work as their own.
- Several people writing one program and turning in multiple copies, all represented (implicitly or explicitly) as individual work.
- Stealing an examination or solution.
- Using unauthorized material during a test or quiz.
- Changing a test, program, or other student work after the work has been graded and requesting that the work be regraded.

DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING
CHEATING POLICY

Not Cheating:

- Turning in work alone or with the help of the course's staff.
- Submission of one assignment for a group of students if group work is explicitly permitted (or required).
- Getting or giving help on any university or department operating system.
- High-level discussion of course material for better understanding. Discussion of assignments for clarification.

DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING
CHEATING POLICY

- When cheating is deemed to have occurred, appropriate disciplinary action will be taken.
- A notice will be placed in the student's permanent computer science record outlining the behavior and the subsequent disciplinary action.
- The instructor may impose a penalty of failure in the course and may deny the student permission to drop the course.
- It is also possible that the student may be barred from subsequent registration in any computer science and engineering courses at the University of North Texas.
- The matter may be referred to the appropriate dean for further university action.

DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING
CHEATING POLICY

- The student is directed to the University of North Texas Student Handbook for general university regulations on cheating under the section on categories of misconduct.
- All procedures concerning cheating are subject to the student's right to due process as outlined in the Student Handbook.

ADA

The Department of Computer Science & Engineering cooperates with the Office of Disability Accommodation to make reasonable accommodations for qualified students (cf. Americans with Disabilities Act and Section 504, Rehabilitation Act) with disabilities. If you have not registered with ODA, we encourage you to do so. If you have a disability for which you require accommodation please discuss your needs with me after class or submit your written Accommodation Request on or before the fourth class day.

Advice

- Get real.
- Don't blow it off. Start early, and finish it before the end of the semester.
- I am not your mother. Don't expect me to nag you.
- You are responsible for getting it done on time.
- Bring group problems to me as soon as possible.
- Don't make me fail you.
- If you can't hack it, get out of the class.

More Advice

- Read your UNT email for messages from me.
- Stay in contact with your group.
  - Exchange email addresses and phone numbers.
  - Answer them.
  - Do not drop off the face of the Earth.
- Meet regularly outside of class times.
- Look at past games at

  `http://larc.unt.edu/demos/`

Summary

- This is a group project class using Subversion, Windows 8.1, Visual Studio 2013, and DirectX 11.2.
- There is a game development lab in NTRP F204.
- There will be homeworks, a milestone, a game contest, and final presentations.
- I'll be teaching by dissecting a collection of game demos for Ned's Turkey Farm.
- I have high expectations.

Summary