Syllabus and Class Overview

CSCE 4210
Ian Parberry
Spring 2017

Class Structure

• There will be 9 lectures (2 hours) & 7 labs (1 hour).
• The rest of the scheduled lecture and lab times will be open development time. Use it wisely.
• Lectures will consist of PowerPoint presentations. Notes will be available on Blackboard. Attendance is expected.
• Labs will consist of a tutorial to be completed during the session. Attendance is mandatory.

About Me

• My name is Ian Parberry.
• If you wish to be formal, call me “Dr. Parberry”.
• Contact: ian@unt.edu.
• For more information: http://larc.unt.edu/ian.

Office Hours

TBA

My Credentials

Why you should listen to what I have to say.

• Professor (since 1984).
• ACM Distinguished Scientist (2015)
• Pioneer of undergraduate game programming education since 1993.
• Credits on 4 commercial games.
• Author of 4 books on game development.
• Author of 36 academic articles on game development research.
1997: SubHunt (Spectrum Pacific)

Programmer.

1997: G-Nome (7th Level)

“Special thanks to…”

2001: TrapWords (MindGames)

Programmer.

2013: TUG (Nerd Kingdom)

Director of Computer Sciences.

My Game Programming Books

2000 2001 2002 2011 2013

Research
Alumni of This Class

• 70 alumni in the game industry since 1994
• 3 alumni authors of game development books
• 3 alumni started game companies
• 3 alumni professors of game development
• 245 credits on 143 commercial games
• 180 million copies sold
• $9B in revenue

Distinguished Alumnus Jason West

Jason West, formerly Infinity Ward (Activision), Respawn Entertainment (Electronic Arts), 1996.

Distinguished Alumnus Art Griffith


Distinguished Alumnus Cesar Stastny

Cesar Stastny, Director of Technology, Treyarch (Activision), 2005.

Distinguished Alumnus Gloria Kennickel

Gloria Kennickel (2006), id Software. Renderer programmer for Rage, voted Best Overall Game and Best Shooter of E3 2010.

Do You Belong In This Class?
This is a tough class. Are you up to it?
Learning Outcomes

1. Be familiar with Windows programming.
2. Be able to use Visual C++ 2015.
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.

Prerequisites

• The listed prerequisite for this class is CSCE 2100: Computing Foundations I.
• You will be dropped from this class if you do not have the official prerequisite.
• 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
  • The C++ standard library

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.

Requirements

These are things you must do.
Musts

- You must work in a group of 2 programmers.
- You must use the class Subversion server.
- You must contribute a significant amount of code to the final game.
- Your game must be playable and bug free.
- Your game must run on the computers in the lab.
- You must pass all of the checkpoints.

Otherwise you will receive a grade of F for this class.

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.

The Revision Control System

- Remember, you must use the Subversion server provided (more details later).
- I will be monitoring your progress during the semester. If you don’t use our Subversion server, I won’t be able to do this. I will assume the worst.
- You must observe Best Practices when using the RCS (more later).
- I will be showing group progress in class towards the end of semester.

Checkpoints

- The labs (required*)
- The pitch (required*)
- The milestone (required*)
- The game contest (required*)
- The final presentation (required*)

*Otherwise you will receive a grade of F for this class.

The Labs

- There will be 7 lab sessions, each of which will have you complete a tutorial.
- Completion of a tutorial during the lab session in which it is assigned will earn you 1 point.
- Completion of a tutorial during the following lab session will earn you ½ point.
- You will lose a letter grade for every 2 points that you miss (I will round in your favor).

Where Will the Labs Be?

- TBA
The Pitch
- Week 5.
- Your group will give a PowerPoint presentation to the class trying to sell them on your game.
- This is called the pitch.
- It is Pass/Fail for the group.

The Milestone
- Week 10.
- Your group will demonstrate your game to the class.
- This is called the milestone.
- It is Pass/Fail for the group.

The Contest
- On the Monday of Week 15 there will be a class game contest.
- Participation is mandatory.
- I will invite judges from the game industry.

The Final Presentation
- Finals Week: May 8-11, 2017.
- Each group will get 30 minutes to show their game and answer my questions.
- All programmers must attend.
- A schedule will be drawn up towards the end of the semester.

My Grading Philosophy
- I grade the executable, not the source code.
- How do I grade a game?
  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 game industry doesn't know how either.
Things I Look For in a Game

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

• Undergraduate students enrolled in CSCE 4210: I expect either completeness or techno points.
• Graduate students enrolled in CSCE 5250: I expect both completeness and techno points.

Completeness points

• Does it run?
• Are there bugs?
• Can you play it?
• Is it fun?
• Is it engaging?
• How far does it go beyond Ned’s Turkey Farm?
• Is there sound?

Techno Points

Awarded for things not covered in class. Examples:
• Better graphics (lighting, bump mapping).
• Level editor.
• Particle engine.
• Procedural content generation.

Your Grade

Your individual grade in this class will depend on:
1. The quality of your group’s game.
2. Your contribution to its executable code.
3. Your correct usage of Subversion.
4. Your group’s performance on the pitch and milestone.
5. Your answers to questions during the final presentation.

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.
• The labs will show you how to build onto this code.
Demos for Ned 2014a

- Demo 0: The Black Screen of Death
- Demo 1: Direct3D
- Demo 2: Scripting and Debugging
- Demo 3: Sprites
- Demo 4: Animated Sprites and Scrolling Background
- Demo 5: Artificial Intelligence
- Demo 6: Scripted Level Design
- Demo 7: Sound
- Demo 8: Playability (Sprite Sheets, Text, and HUD)
- Demo 9: Pixel Shaders
- Demo 10: Two-player networking
Why Teach This Way?

- Once you know how to program, the best way of learning is by *reading and doing*.
- 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 online.
- It is, however, large and complex.
- Getting started is the hardest thing.
- I 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.

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.

Schedule

Put this stuff on your calendar. Now.
Lectures
1. Syllabus, Group Formation
2. Game Design
3. Subversion, Visual Studio 2015, Demo 0
4. 3D Math, DirectX 11.2, Demo 1
5. Demo 2, Demo 3
6. Demo 4, Demo 5
7. Demo 6, Demo 7
8. Demo 8, Demo 9
9. Demo 10

Labs
1. Game Design
2. Demo 0: Windows programming
3. Demo 1: Top down games
4. Demo 3: Platformers
5. Demo 5: AI
6. Demo 7: Sound
7. TBA

Calendar

<table>
<thead>
<tr>
<th>Date</th>
<th>Lecture</th>
<th>Lab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 23</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Jan 30</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Feb 6</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Feb 13</td>
<td>Pitch</td>
<td>Dev</td>
</tr>
<tr>
<td>Feb 20</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Feb 27</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Mar 6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Mar 20</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>Mar 27</td>
<td>Milestone</td>
<td>Dev</td>
</tr>
<tr>
<td>Apr 3</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>Apr 11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apr 17</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Apr 24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 1</td>
<td>Content</td>
<td>Dev</td>
</tr>
<tr>
<td>May 8-11</td>
<td>Presentations</td>
<td>Dev</td>
</tr>
</tbody>
</table>

Official Class Policies
These are not negotiable.

Cell Phone Policy
- Please mute your cellphones.
- However, having a cell phone ring in class is not the end of the world.
- Please put them away during my lectures.
- Similarly for computers, unless you are working on your game.

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 University of North Texas complies with Section 504 of the 1973 Rehabilitation Act and with the Americans with Disabilities Act of 1990. The University of North Texas provides academic adjustments and auxiliary aids to individuals with disabilities, as defined under the law. Among other things, this legislation requires that all students with disabilities be guaranteed a learning environment that provides for reasonable accommodation of their disabilities. If you believe you have a disability requiring accommodation, please see the instructor and/or contact the Office of Disability Accommodation at 940-565-4323 during the first week of class.
My Advice To You
Act like a grownup. Really.

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
• Pay attention to Blackboard (learn.unt.edu).
• 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/

In Conclusion

Summary
• This is a group project class using Subversion, Windows 10, Visual Studio 2015, and DirectX 11.2.
• There will be tutorials, 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.