

# COMP7300 – Advanced Games Programming

Semester 1, 2004

**Lecturer:** Gordon Wyeth, Room 78-626, (until mid-semester break).

**E-mail:** wyeth@itee.uq.edu.au

**WWW:** <http://www.itee.uq.edu.au/~comp7300>

**Phone:** 3365 3770

## **Purpose/Goal:**

This course deals with state of the art programming techniques for the development of interactive computer games. Topics focus on current relevant programming environments for computer gaming, and many of the relevant technologies associated with computer gaming (such as graphics, sound, input devices, AI and simulation). After completing this course, you will be aware of many of the issues associated with games programming, and will have mastered a significant skill base relevant to implementation of computer games.

## **Credit: 2 Units**

**Pre-Requisites:** COMP2301 is highly recommended. Students in the course are expected to have substantial previous experience with Windows programming using C or C++ on entry. A solid knowledge of mathematics is expected, as well as knowledge of basic Newtonian physics.

**Incompatible:** - .

**Workshops:** Monday 9am – noon and 1pm – 3pm.

**Assessment:** Project based, details below.

**Textbook:** Andre LaMothe, "Tricks of the Windows Game Programming Gurus", Second Edition, 2002, ISBN 0-672-32369-9 is the compulsory text. It is available in the library at call number QA76.76.C672 L3615 2002.

## **0.1 Resources**

### 0.1.1 Course Structure

The course is divided into a number of learning Modules that cover specific skill sets required to complete this course. The content and order of the Modules is described below. The Modules will be the focus of the first 7 weeks of the Semester. The subsequent time will focus on the production of the computer game that will form the basis for your assessment.

### 0.1.2 Study Guides

Before each of the Modules is presented, I will provide a set of study notes for that Module – the study guide. The guides will only be available from the subject WWW home page. They will provide a basic coverage of the content for the module, as well as providing any scanned material and sometimes a list of further readings. The notes will reference the textbook and any scanned material in context, with a reading symbol to indicate that you should address that material at this point in the guide. The study notes will also contain questions and exercises throughout, some of which will be in the text. You should perform these exercises before you come to class. If you do not prepare for the class you will not get the full benefit from the workshops.

Some exercises in the Study guides are marked for hand-in. Be sure to keep your solutions to these exercises on disk for assessment purposes.

The study guides are **not** a complete set of lecture notes. They are simply there to highlight material in the text and to bring in any other material in context.

### 0.1.3 Textbook

Andre LaMothe, "Tricks of the Windows Game Programming Gurus", Second Edition, 2002, ISBN 0-672-32369-9 is the compulsory text. It is available from the University bookshop and many book retailers. Several copies are available in the library at call number QA76.76.C672 L3615 2002.

### 0.1.4 Other books

Other useful references are:

Ian Parberry, "Introduction to Computer Game Programming with DirectX 8.0" is available from many book retailers.

### 0.1.5 Workshops

I will spend very little of our time together talking – apart from introducing the material in the study guide and the relevant sections of the text. The mornings will be used to discuss and reflect upon the study guide and text. We will do this by discussing the questions and exercises set in the study guide. We can then explore the themes that groups discussed across the whole class. The afternoons will be run by setting a single largish problem(s) to be solved in the afternoon. You will work in groups and present your solutions and thoughts to the class.

## 0.2 Administrivia

### 0.2.1 Lecturer Availability

I will be available for the ten minutes before and after each class for brief questions. This is the best time to see me about COMP7300. If you require a longer meeting, this is the best time to arrange one with me. If you are unable to contact me in person, send me email.

Apart from this time, I have an open door policy that allows student consultation (on any matter) at all times. If I am in my room, simply come to the door and knock.

Otherwise, my presence on campus can be determined by a simple system. If the light is on and my door is open then I am somewhere nearby and will return shortly. If the light is on but the door is closed, then I am on campus and will be back in my room at some stage during the day. If you come to my room and I am in consultation with another student or staff member, please wait until I have finished with that meeting before knocking.

Please understand that I have a number of roles in the department (fourth year student supervision, postgraduate student supervision, research, writing publications, collegial meetings, departmental meetings, robot contests, lab supervision, course advising, curriculum development etc.) and I may not be always be in my room or able to answer your question on the spot. If you need a meeting, see me at the lecture or send me email.

### 0.2.2 Subject Announcements

Please read the newsgroup `uq.csee.eng` on a regular basis for announcements about COMP7300. All announcements will have COMP7300 in the header. This is also a good forum for general discussion or questions about the subject. Please include COMP7300 in the header. Anonymous postings will be disregarded.

### 0.2.3 Collection and Distribution of Material

Study guides may be downloaded from the WWW in PDF format. The solutions to exercises should be submitted using the on-line submission facility. Written game proposals will be handed in during the class session in which they are presented. The assessed proposal will be returned the following week in class. The final game (including the source, binaries and documentation) will be handed-in in the examination period, and will be available for return from my office after the release of results.

### 0.2.4 Academic Merit, Plagiarism, Collusion and Other Misconduct

The School and the wider academic community in general takes academic integrity and respect for other persons and property very seriously. In particular, the following behaviour is unacceptable: Submission of plagiarised work, i.e. work that contains content copied from an unacknowledged source. Submission of work without academic merit, i.e. work that adds little or nothing to material available from reference sources such as textbooks, websites, etc., even where this is appropriately acknowledged. Engaging in collusive behaviour, i.e. inappropriate working together with other students where individual work is required, or working with people outside your team where team work is required. Copying work done by other students. Failing to adhere to the School's regulations concerning behaviour in laboratories, in particular occupational health and safety regulations.

Penalties for engaging in unacceptable behaviour can range from cash fines or loss of grades in a subject, through to expulsion from the University. You are required to read and understand the School Statement on Misconduct, available on our website at: <http://www.csee.uq.edu.au/about/student-misconduct.jsp> . If you have any questions concerning this statement, please contact your lecturer in the first instance.

### 0.2.5 Disability Action Plan

Any student with a disability who may require alternative academic arrangements in the course is encouraged to seek advice at the commencement of the semester from a Disability Adviser at Student Support Services.

## 0.3 Expected Outcomes

In general terms, you are expected to demonstrate that you have the necessary skills to write an engaging computer game in a DirectX environment.

In more specific terms, I intend for you to accomplish the following by the end of this course:

### **Module 0: Introduction to Game Programming**

- to understand what constitutes a game, what makes a successful game, and what makes a successful games programmer;

### **Module 1: Windows Programming**

- to thoroughly understand the Windows environment particularly notation, classes, events, resources and GDI (review material),
- to be able to use COM objects;

### **Module 2: DirectX and Basic DirectDraw**

- to be able to use DirectX COM objects,
- to be able to use DirectDraw to make surfaces, draw on surfaces and manipulate colour,

### **Module 3: 2D Graphics in DirectX**

- to master games programming techniques for 2D graphical environments under DirectX,

### **Module 4: DirectInput**

- to create user input using the DirectInput features of DirectX;

### **Module 5: Sound and Music**

- to be able to generate sound effects and music for a computer game environment;

### **Module 6: Artificial Intelligence**

- to understand the basic AI techniques used in agent programming in computer games,
- to be able to code an AI agent for a computer game;

### **Module 7: Simulation and Modelling**

- to understand the techniques used to model physical laws in computer games,
- to understand the principles of kinematic equations for describing motion.

## **0.4 Assessment**

The first component of assessment is to ensure that you complete the exercises in the Modules. The other components of assessment are based around a single project that involves the design and implementation of a complete, playable game demo of your choice. There are two assessment phases for the project – the assessment of the game proposal, and the assessment of the game itself.

### **0.4.1 Module Exercises (20%)**

The module exercises will be assessed in Week 11. You will demonstrate working binaries and hand in all of the source for the exercises that are marked for hand-in in the Modules. You will be evaluated on the demonstrability and completeness of the work. Your source code will be checked for readability, maintainability and style. Detailed criteria will be provided during class. The work will be graded on a 1 - 7 scale.

### **0.4.2 Game Proposal (30%)**

The game proposal will be assessed in Week 10. The proposal will consist of both oral and written presentations. The oral presentation will take place during the week 10 workshop. You are to provide your written report on your game proposal to me at the start of your oral presentation. You will be assessed on how thoroughly you have thought through the design of your game, and how well you have planned its implementation. Detailed criteria will be provided during class. The work will be graded on a 1 - 7 scale.

### **0.4.3 Final Game (50%)**

The game itself will be assessed over the examination period. A provisional assessment of your game will be performed in the final week of semester to give you some direct feedback on your game before final assessment. The final game (including the source, binaries and user manual) will be handed-in in the examination period at the time designated in the exam timetable – you can think of it as a semester-long take home exam. The principle criterion for assessment purposes is how well your game plays. To this end, I will be principally concerned with your binaries and your user manual. Your source code will be checked for readability, maintainability and style. Detailed criteria will be provided during class.

### **0.4.4 Determination of Final Grade**

The final grade will be determined by rounding the sum of the weighted grades from the three assessment components.

## 0.5 Syllabus

Week	Starts	Content
1	1/3	<b>Module 0:</b> Introduction to Games Programming
2	8/3	<b>Module 1:</b> Windows Programming
3	15/3	<b>Module 2:</b> DirectX and Basic DirectDraw
4	22/3	<b>Module 3:</b> 2D Graphics in DirectX
5	29/3	<b>Module 4:</b> DirectInput <b>Module 5:</b> Sound and Music
6	5/4	<b>Module 6:</b> Artificial Intelligence
7	19/4	<b>Module 7:</b> Simulation and Modelling
8	26/4	<b>Anzac Day - Holiday</b>
9	3/5	<b>Labour Day - Holiday</b>
10	10/5	<b>Game Proposal</b>
11	17/5	<b>Hand-in Exercises</b>
12	24/5	<b>Game Development</b>
13	31/5	<b>Provisional Assessment</b>

## 0.6 Game Design

In this section we're going to talk a little about game design. Ultimately, designing games is something that you have to **do** to understand – and you'll be doing that in parallel with the regular assignments over the first seven weeks.

### 0.6.1 History (My Personal Spin)

There has been a lot of games research and development over the years which make a good starting point for a discussion of game design. Before the personal computing paradigm took off, most games were based around text interaction with the computer or with other users on the computer. The emphasis here was on AI rather than modelling and graphics. There were quite a few papers written about this stuff and a lot of the source still floats around today. In the late '70s the biggest game market was the arcade game with the arrival of Space Invaders *et al.* This was arguably the start of the PC revolution people buying TRS-80, Apple, Atari so that they could play games at home. In order to justify the expense of the machine, the designers of PCs

were forced to come up with other *really useful* applications – such as recipe storage. A number of companies recognised the importance of games and developed the best games hardware of the day, but neglected to provide the justification for expense in terms of unfun applications. When the IBM PC and clones came out, they weren't real good at games but the justification was there, and people bought them. Games developers then started working out how to make them play. As the PC hardware improved and its market base increased it basically took over as the games platform. Nowadays, of course, PCs have super fast processors, incredible graphics engines, awesome sound generation ability and a range of input devices. They are the number 1 platform for games in terms of market share and are still growing over specialist competitors like Sony and Nintendo (that whole justification thing again). Nevertheless the specialists are now fighting back by giving their boxes the ability to play DVDs and surf the web.

### 0.6.2 A Skeleton Game Design Process

Here's an outline of the basic steps to designing a computer game:

1. Choose a genre. Broadly speaking you have:
  - First Person Shooters
  - Sports
  - Fighting
  - Arcade
  - Simulation
  - Strategy
  - Storybook
  - Board
2. Get creative. Generate as many ideas as you can and record them. Don't dismiss any as you go along – just let them all come out. Strange ideas are sometimes stepping stones to really cool ideas. Once you have your list, then start to cut back those that are just not really going to work, and work through some of the better ones to see whether they will work. This process might need to be repeated several times to get something you can go on with.
3. Write a design document. Before you code anything you should write a close to definitive design document that explains as many details about the game as possible. Imagine that you know nothing about computer programming, and want to explain your idea to somebody who does in order to get it produced. In other words your document should fully explain the function of the game, not the implementation.
4. Develop a plan. Now that you know what you want, start to work out how it can be done. Importantly check that it *can* be done in your budget and time frame. If it can't be done, then its time to re-visit step 3 and review the design.

5. Do an audience test. Once you have come up with a feasible design, run it past some people that haven't heard your ideas. Then ask them the all important questions – Is this game cool? How much download quota would you give up to play this game? (Asking a cash value on your first game might be a bit ambitious.)

We will be working through this procedure in class for your game design for this subject.

## 0.7 Components of a Game



The text describes the general layout of a game in terms of its key software components. Have a read of pages 17 – 21, paying particular attention to the code listing.

## 0.8 Game Programming Tricks



The text (pages 21 – 27) has a long list of game programming tricks that relate to any programming exercise, particularly time-critical programs.

## 0.9 Getting Started



*Look at the Freakout code listing (pages 31-45). List which functions fit into which game components.*

*Compile the Freakout game under VC++.*



*Add a feature to the game that increases the speed of the ball as your score increases, and decreases the ball speed as your score decreases.*

## 0.10 Preparation for Next Week



Read Chapters 2, 3, 4 and 5. That would be pages 47-238. There'll be plenty of exercises on this material in the next module.