Lecture 1: Introduction and Preliminaries
Teaching Staff and Help

Dr Rowan Davies (Rm 2.16, opposite the labs)
rowan@csse.uwa.edu.au

Help: via help4242, project groups, during labs, during and after lectures.

Note: You will be given some credit for answering questions and posting useful information on help4242 when the project is assessed.
Computer game technology draws expertise from several different areas including graphics, physics modelling, agent technologies and artificial intelligence. This unit provides an overview of the key issues of game design, such as playability, game play, the target audience and the various genres of computer games. The core of the unit explores three-dimensional modelling, animation, and rendering of computer graphics. It also introduces Newtonian physics-based modelling of interactions between characters, items (e.g. weapons, vehicles) and the virtual environment. Techniques from agent-oriented software engineering and artificial intelligence are investigated to assist advanced user interaction and decision making during the game play. Integrated leading-edge game development environments and libraries are used for practical implementation of the key components of a computer game.
Assessment

This comprises a programming project (60%) and a final examination (40%).

The project involves problem identification, formulation and implementation in relation to creating a three-dimensional single- or multi-player game.

The examination tests in-depth technical competence in issues related to game design.

Prerequisites: CITS2231 Graphics
Lecture Topics

Last year the following were covered:

• Lecture 1: Introduction and Preliminaries
• Lecture 2: Game Design
• Lecture 3: Game Engines and Game Content
• Lecture 4: 3D Modelling with Subsurfaces
• Lecture 5: C# Essentials
• Lecture 6: Game Entity Architecture and NeoAxis Tutorial
• Lecture 7&8: UV Unwrapping and Animation
• Lecture 9&10: Physics
• Lecture 11: Scenes, Particles and Normal Mapping
Lecture Topics

• Lecture 12: Intro. to Game AI
• Lecture 13: Finite State Machines for Games (PPT)
• Lecture 14: Steering Behaviour (Part I)
• Lecture 15: Guest Lecture
• Lecture 16: Steering Behaviour (Part II) Movie and the BigShoal Group Behaviour
• Lecture 17: Group AI (PPT)
• Lecture 18: To Script or Not to Script
• Lecture 19: Graphs in Game AI
• Lecture 20: Practical Path Finding
• Lecture 21: Goal-Oriented Behaviours (PPT)
• Lecture 22: Fuzzy Logic in Games
• Lecture 23: Hidden Markov Model in Games
The Project is the main focus in this unit

- It is worth 60% of your mark.
- It will be handed out in week 2, and due week 13.
- You will work in groups of roughly 4.
- Each group can choose what they build.
- You will be assessed on submitting an individual portfolio that includes details of your contributions.
- In weeks 4, 6, 8, 10, 12 & 13 each group will be required to briefly show their progress to the whole class during the lab session.
We will use the following main tools:

- **NeoAxis game engine and associated tools**
  - Includes rendering based on the open source Ogre 3D engine.
  - Also makes use of popular libraries for physics, sound, etc.
  - Tools are included for map editing and resource editing.
  - Uses C#, which is similar to Java.
  - A wide range of examples are included with source code.

- **Blender open source 3D modelling and animation software**
  - Textured models and animations can be exported and used by NeoAxis.

- **Subversion version control system**
  - To track changes to a project and allow many people to work on different parts.
Labs

There will be some set labs designed to teach you specific skills.
- In the first few weeks: NeoAxis, Blender, C# and Subversion.
- Later: Game logic and AI

The remainder of the lab time is for you work on the project.

Lab attendance is compulsory in weeks 2, 4, 6, 8, 10, 12, 13.
- This is to form groups (week 2) and to learn from other groups experiences.
- If you cannot attend, please email ahead of time to explain (otherwise your mark may be affected).
- If you have a clash, or will often find it difficult to attend, tell me.
What does it take to develop a game?

Game development requires many quite different activities.

- Game design
  - Developing the basic concept of the game.
  - Developing a background story for the game and the characters.
  - Designing the basic game mechanics.
  - Deciding on a general “feel” or mood for a game. E.g., humorous, sombre, or mystical.

- Designing graphics and animation
  - Often in 3D, using modelling software such as Blender (or 3Dstudio or Maya).
  - In practice this is one of the largest tasks for major commercial games.
What does it take to develop a game? (cont)

- Programming
  - Building a game engine.
    - Real-time graphics
    - Sound, I/O
    - Physics
  - Building tools and general code to hold everything together.
  - Coding the logic of a game: e.g. “The player can only open a door if they have a key.”
  - Coding the AI for characters in the game.

- Project management
  - Generally game development requires a highly iterative development model.
    - The main requirement is “be fun” - often this can only be judged after building something.
  - It is important to make sure each iteration is a step forwards, not in circles.
  - It is important to keep the people involved “on the same page”.