Architecture of Computer Games

SE 456

Instructor: Ed Keenan
Email: ekeenan2@cdm.depaul.edu
office hours: Mon / Tuesday 4-5pm, 9-10pm , or by appt
office: CDM 830
phone: (312) 362-6747

Description:

Learn the principles, styles, and patterns of software architecture and framework design in the context of computer game development. The class is focused on the construction and design of a game using supplied frameworks and game engines. Learn the working components of a game, game loop, graphics, collisions, audio system, input controls and state machines. Emphasis of the macro/micro design patterns will be incorporated to control the complexity and efficiency of the game design. Design issues and challenges of 2D and 3D games will be explored through various APIs.

This is intensive software development course using the C# - XNA windows environment. Students will work on understand the requirements, design, and implementation of individual game components of a complete game. A final project of developing a complete computer game will be emphasized throughout the class.

Software engineering aspects throughout the class will be emphasized. You will learn software design principles such as design pattern and software architecture styles that can be applied in the design of object-oriented frameworks for game development, i.e., object-oriented game engines. We will explore the trade-offs and ramifications of software architecture and design choices with respect to performance, maintainability, and reusability, etc. We will also discuss how some of the best practices in software engineering such as, iterative development, testing, and continuous integration, can be applied in game development.

Prerequisites:

- Advanced knowledge of Java or C++
- Data Structures (CSC 383 or CSC 393)

Grading

45% - Programming Assignments
  - Input, Sprite, Collision, Shield, Sound systems
  - Code and video demo for each assignment

10% - Perforce Submissions / Piazza Participation

40% - Final Project (Design Document, Complete Game, Demo)

5% - Final paper (Reflective)

Textbooks and printed resources

Additional course material will be many supplied through class notes, handouts or online links.

- Lecture notes and supplementary materials provided by the instructor.
- Additional material
    - Design Patterns Resources
        - Head First Design Patterns by Freeman
        - Design Patterns: Elements of Reusable Object-Oriented Software by Gamma, Helm, Johnson and Vlissides
    - XNA
        - Learning XNA 4.0 by Reed

Topics will include:

Game Fundamentals

- Game Loops
- Timers
- Audio
- State Machines
- Input

2D Games

- Sprites
- Collision
- Movement
- Images loading / manipulation

Design Patterns

- Singleton
- Factory
- Null Object
- Flyweight
- State
- Strategy
- Observer – Publisher/Subscriber
- Memento
- Data Driven

Project Management

- Iterative Development
- Scheduling

Programming Assignments – 45%

Programming assignments are broken into: 5 programming assignments. Each assignment builds on the previous one. Each deliverable will be evaluated according to the sub-goals. With every assignment there will be a video capture and detailed engineering documentation.

- ***PA 1:*** (5%) Input Manager
  - Input recorder / manager
  - Ability to search for specific button combos
  - Clean abstraction and mapping
  - Get familiar with XNA and perforce

- ***PA 2:*** (10%) Sprite / Animation system
  - Create a sprite system that displays the sprite image
  - Animate the sprite to display an different image at a time
    - Set the series of images to cycle through
    - Set the timing / speed to display the images
  - Ability to reuse sprite images instead having duplicate images loaded at the same time.
  - Should be general enough to display any sprite used in the game
    - Alien Ships, Shields, Player, UFO, missiles

- ***PA 3:*** (10%) Collision system
  - Create a collision system that determines collisions between the missiles and collideable objects
    - Such as the Aliens, Player, Ship, UFO, Collision subpart in the shield
  - The collision system should be able to determine the if any missile hit the collision box.
    - Ideally this system should be able to intersect with missile box with the target box
    - Determine the state of the collision, {non-intersection, or intersection}
  - This system should be able to be used with any collision needs in this game:
    - Aliens, Player, Ship, UFO, Collision subpart in the shield

- ***PA 4:*** (10%) Shield collision and display system
  - Determine how to have the shield be dissolving and eroding by missiles from the aliens
  - Determine how the shield is dissolved and eroded by missiles from the player
  - Shield as a protector
    - It protects the player, from the alien missiles
  - Drawing mechanism to show the erosion
  - Determine the underlining collision grid and update mechanism

- ***PA 5:*** (10%) Sound system
    - Create a system that loads and plays static waves in the game
    - Allow other systems to call single of sound waves for their respective effects
    - Ability to control individual volumes of each sound playing
    - Allow many different sounds to be playing at once,
        - i.e. multiple explosions overlapping
    - Gross sound controls
        - Mute, game overall volume

Final Project – 40%

Students will design and build a clone of the arcade version of Space Invaders. This cloned game will be written in C# using the XNA framework using MODERN software engineering principles, including design patterns, iterative based development, robust and orthogonal systems.

- Stand alone Game application will include:
    - Select Screens,
    - Game cycling
        - (select->enter->game over->select),
    - Score,
    - Player movement and control,
    - Alien Grid,
    - Animation,
    - UFOs,
    - Missiles,
    - Bombs,
    - Erodible shields,
    - graphics effects
    - Sound
- Design Documentation
    - UML diagrams of all the major systems
    - Descriptions of the interactions between components
    - This will be written a little bit every week 3-5 pages per major system
- Video demo of the game working with commentary
    - Demonstrating every featured aspect of the game
    - Show the game running and playing across several screens and games

Final Paper – 5%

Students will write a Reflective about the challenges and difficulties in design, implementation

Perforce Submissions & Piazza Participation - 10%

Perforce Submissions

- Every is expected to submit at least 10 submissions a week to perforce.
- The biggest reason students get into trouble with software design:
    - Not working on the material frequently enough
    - Taking too large of a bite of the design
- Both are fixed with this Perforce RULE.
- Even my simplest programs take 10-20 submissions.
    - For these project assignments my average is 40 submissions, so 10 will be no problem.

Piazza Discussion forum

- Statistics show: students who participate more and help other students do better!
    - The correlation is ridiculous!
        - Poor understanding / poor participation.
        - Great understanding / Great participation
    - As you master the material, help others learn!
        - You're in the master's program so master it!
- Everyone is expected and encouraged to participate on the Piazza discussion forum. All class-related discussion here this term.
- The quicker you begin asking questions on Piazza (rather than via emails), the quicker you'll benefit from the collective knowledge of your classmates and instructors. We encourage you to ask questions when you're struggling to understand a concept.
- All correspondence that is not personal in nature should be vectored through Piazza
- Sensitive material, use Piazza private channel.

Collaborating together on programming assignments

- You are encourage to work together
    - Use the Piazza forums heavy
    - Even share your material with others in the common directory
- Everyone is 100% responsible for the work they do.
    - If you get help with a section of code,
    - Please refactor the code
        - Comment and understand that material
        - Transform the code to *make it yours*.
- System for Detecting Software Plagiarism
    - We will be using MOSS - Measure of Software Similarity (Standford University)
        - Indicates possible code infringements (plagiarism)
        - MOSS - will detect the similarity independent of naming convention, indentation style or formatting, it compares abstract syntax tree of your code.

- If you gain significant support / help from another student
    - Fully disclose the support / help you had in a Readme.txt file submitted with your assignments.
        - Disclosing the help, is not permission for copying the code.
        - Only there to clarify and acknowledge help you were given from a fellow student.
- Modifying any Unit Test to alter the outcome results is also an Academic Integrity Violation
- If you are stuck and find yourself even tempted to plagiarize
    - Ask for help !!!!
        - Use on Piazza,
        - Visit during offices hours, make an appointment
    - Don't compromise your integrity!

Schedule:

Week	Lecture	Assign	Due
1	Overview, wiki, perforce UML, C#, XNA Input Manage	Input Manager	
2	*Patterns*: Singleton, Factory, Flyweight Sprite System Design Reviews / Checkin process	Sprite / Animation	
3	*Patterns*: Composite, Observer/ Callbacks Collision System	Collision System	Input Manager
4	*Patterns*: Command, Iterator Shield System Test System	Shield System	Sprite / Animation
5	*Patterns*: Queues, Null Object Sound System	Sound System	Collision System
6	*Patterns*: Strategy, State, Momento Game Design - putting it together		Shield system
7	*Patterns*: Prototyping Design Documents	Design Docs	Sound System
8	*Patterns*: Adaptors, Proxy, Façade Future proofing your code Changing Game engines		Design Docs
9	Middleware solutions Engineering in Games		
10	Space Invaders trouble shooting Cost / Business of Games		
11			Final Project

School policies:

Online Instructor Evaluation

Evaluations are a way for students to provide valuable feedback regarding their instructor and the course. Detailed feedback will enable the instructor to continuously tailor teaching methods and course content to meet the learning goals of the course and the academic needs of the students. They are a requirement of the course and are key to continue to provide you with the highest quality of teaching. The evaluations are anonymous; the instructor and administration do not track who entered what responses. A program is used to check if the student completed the evaluations, but the evaluation is completely separate from the student's identity. Since 100% participation is our goal, students are sent periodic reminders over two weeks. Students do not receive reminders once they complete the evaluation. Students complete the evaluation online at https://mycti.cti.depaul.edu/mycti

Email

Email is the primary means of communication between faculty and students enrolled in this course outside of class time. Students should be sure their email listed under "demographic information" at http://campusconnect.depaul.edu is correct.

Academic Integrity Policy

This course will be subject to the academic integrity policy passed by faculty. More information can be found at http://academicintegrity.depaul.edu/

Plagiarism

The university and school policy on plagiarism can be summarized as follows: Students in this course should be aware of the strong sanctions that can be imposed against someone guilty of plagiarism. If proven, a charge of plagiarism could result in an automatic F in the course and possible expulsion. The strongest of sanctions will be imposed on anyone who submits as his/her own work any assignment which has been prepared by someone else. If you have any questions or doubts about what plagiarism entails or how to properly acknowledge source materials be sure to consult the instructor.

Incomplete

An incomplete grade is given only for an exceptional reason such as a death in the family, a serious illness, etc. Any such reason must be documented. Any incomplete request must be made at least two weeks before the final, and approved by the Dean of the College of Computing and Digital Media. Any consequences resulting from a poor grade for the course will not be considered as valid reasons for such a request.

Resources for Students with Disabilities

Students who feel they may need an accommodation based on the impact of a disability should contact the instructor privately to discuss their specific needs. All discussions will remain confidential.

To ensure that you receive the most appropriate accommodation based on your needs, contact the instructor as early as possible in the quarter (preferably within the first week of class), and make sure that you have contacted either:

- PLuS Program (for LD, AD/HD) at 773-325-4239 in SAC 220
- The Office for Students with Disabilities (for all other disabilities) at 773-325-7290 Student Center 307