

# Gaming Architecture & Programming

B.E. Sem. VIII [INFT]

---

---

## EVALUATION SYSTEM

	Time	Marks
<b>Theory Exam</b>	3 Hrs.	100
<b>Practical Exam</b>	–	–
<b>Oral Exam</b>	–	25
<b>Term Work</b>	–	25

## SYLLABUS

- **Prerequisite:** Proficiency in C/C++ programming.
- **Objective:** An important characteristic of technical education is an emphasis on their challenging nature, the structured character of the concepts, the critical role of quantitative problem solving, and the importance of qualitative reasoning.

Much of the difficulty in mastering technical subjects lies in the importance of abstract variables and the constraints among them. Concepts of Modern game software architecture, Game worlds and game objects, Collision detection, Events and scripting, Introduction to animation and Implementing game play would help students acquire this knowledge. This course provides students with an introduction to the technologies and software engineering practices used in the video game industry today. Students will learn the basics of creating a PC game based DirectX, through lecture material, hands-on labs, and a final project in which the students will actually build a simple game from the ground up.

### Game Architecture

1. **Core Design:** What Is a Game? Games Aren't Everything. Games Mean Gameplay. Creating the Game Spec. Example Game Spec.
2. **Initial Design:** The Beginning. Hardware Abstraction. The Problem Domain. Thinking in Tokens.
3. **Use of Technology:** The State of the Art. Blue-Sky Research. Reinventing the Wheel. Use of Object Technology.
4. **Building Bricks:** Reusability in Software.
5. **Initial Architecture Design:** The Birth of Architecture. The Tier System. Architecture Design.
6. **Development:** The Development Process. Code Quality. Coding Priorities. Debugging and Module Completion. The Seven Golden Gambits. The Three Lead Balloons.

### Game Programming

7. **Technologies:** Display, Mixing 2D & 3D, DirectX, User Interface code, Resource caching, the main loop.
8. **Design Practices:** Smart & naked pointers, using memory correctly, Game scripting languages.

9. **Building your game:** Creating a project, source code repositories and version control, Building the game and scripts.
10. **User interface programming and input devices:** Getting the Device State, Working with the Mouse (and Joystick), Working with the Keyboard, User Interface Components, More Control Properties.
11. **2D Drawing and DirectX:** 2D Drawing and DirectX, Basic 2D Drawing Concepts, Drawing Text, Working with Sprites, Graphics File Formats.
12. **Initialization and the Main Loop:** Initialization, Some C++ Initialization Pitfalls, Initializing your Game, the Main Loop, Stick the Landing: A Nice Clean Exit.
13. **Loading and Caching Game Resources:** Art and Sound Formats, Resource Files, Data Compression, IPac: A Resource File Builder, the Resource Cache, World Design and Cache Prediction.
14. **3D Graphics & 3D Engines:** 3D Graphics Pipeline, Setting Up a Project, Using a Scene Graph, 3D Middleware Review, Rolling Your Own 3D Engine.

**Reference Books:**

1. Game Architecture and Design, (*Andrew Rollings & Dave Morris*)
2. Professional Game Programming (*Mike McShaffry*) Dreamtech Press.
3. Game Programming, (*Andy Harris*) Wiley India.

