DigiPen-JTech Game Development Academy

Level 2

Welcome to Level 2 at DigiPen-JTech Game Development Academy! Here, you'll dive deep into 2D game development. You'll learn core concepts, create exciting projects, and sharpen your programming skills.

Core Concepts in Game Development

1

Game Objects and Structures

Efficiently define and manipulate game entities using custom data structures to represent their properties and behaviors.

2

Arrays and Loops

Store and access multiple game objects using arrays, and leverage loops to perform repetitive actions on them.

3

Functions and Conditional Logic

Break down complex game logic into reusable functions, and make decisions based on specific in-game conditions.

4

Input/Output and Game Physics

Handle user input (keyboard, mouse) and display game information, while simulating realistic physical behaviors like gravity and collisions.

5

Artificial Intelligence

Create intelligent game agents that can make decisions and react to the player's actions, adding depth and challenge to the gameplay.

Game Objects and Structures

Define

Create game objects using structures to organize data efficiently.

Manipulate

Learn to modify object properties dynamically during gameplay.

Interact

Implement object interactions to create engaging gameplay mechanics.

Optimize

Manage memory usage by designing efficient object structures.

Arrays and Loops in Game Development

Arrays

Store multiple game objects efficiently. Access elements quickly for smooth gameplay. Use multi-dimensional arrays for complex game states.

Loops

Repeat code blocks for game updates. Iterate through arrays to process game objects. Create game loops for continuous gameplay.

Synergy

Combine arrays and loops for powerful game logic. Optimize performance by reducing redundant code. Create dynamic, responsive game worlds.

Functions and Conditional Logic

1

Define Functions

Create reusable code modules for game mechanics. Improve code organization and readability.

2

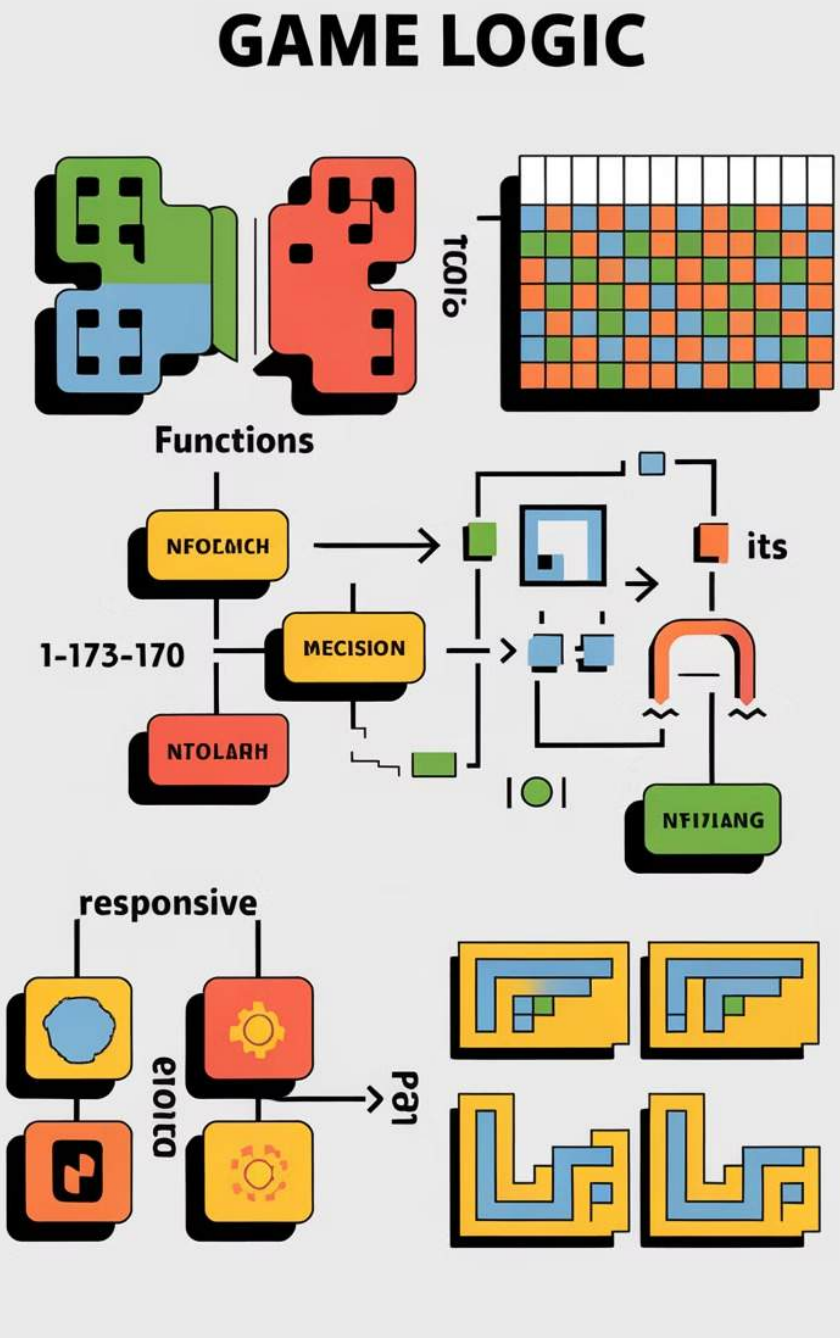
Implement Conditional Logic

Use if-else statements to make game decisions. Create branching paths in your game flow.

3

Combine Functions and Logic

Build complex game systems with modular, decision-making code. Enhance gameplay with dynamic, responsive behavior.



Input/Output and Game Physics

Keyboard Input

Capture player actions for responsive controls. Map keys to game functions.

Mouse Input

Implement clicking and dragging mechanics. Create intuitive user interfaces.

Physics Simulation

Apply gravity and collision detection. Create realistic object interactions.

Game Output

Display scores, health, and game states. Provide clear feedback to players.

Artificial Intelligence in Games

1

Decision Making

Implement AI algorithms for intelligent game agents. Create challenging and engaging opponents.

2

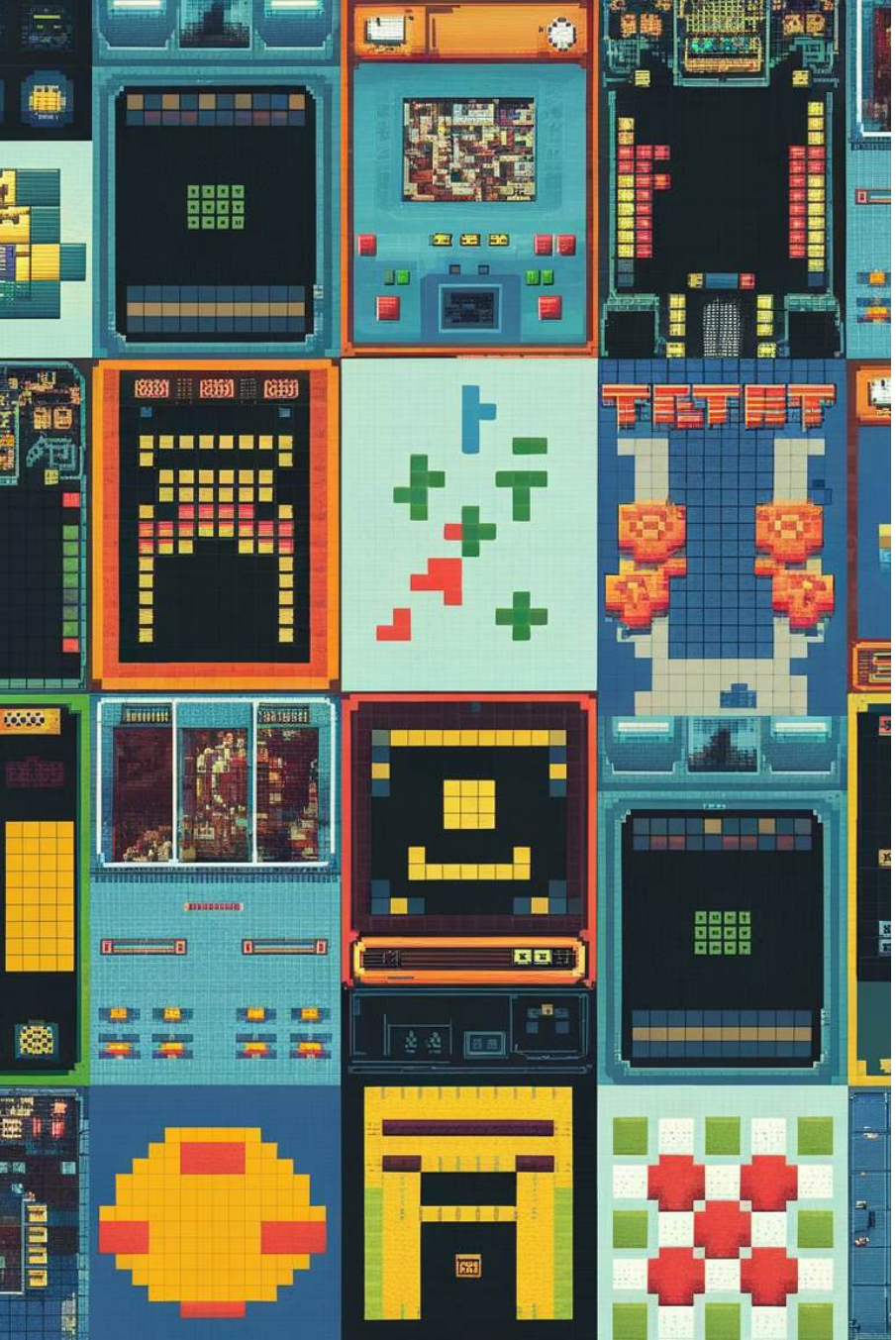
Pathfinding

Develop AI that can navigate game environments. Implement A* algorithm for efficient movement.

3

Behavior Trees

Design complex AI behaviors using hierarchical structures. Create dynamic and responsive game characters.



Game Development Projects

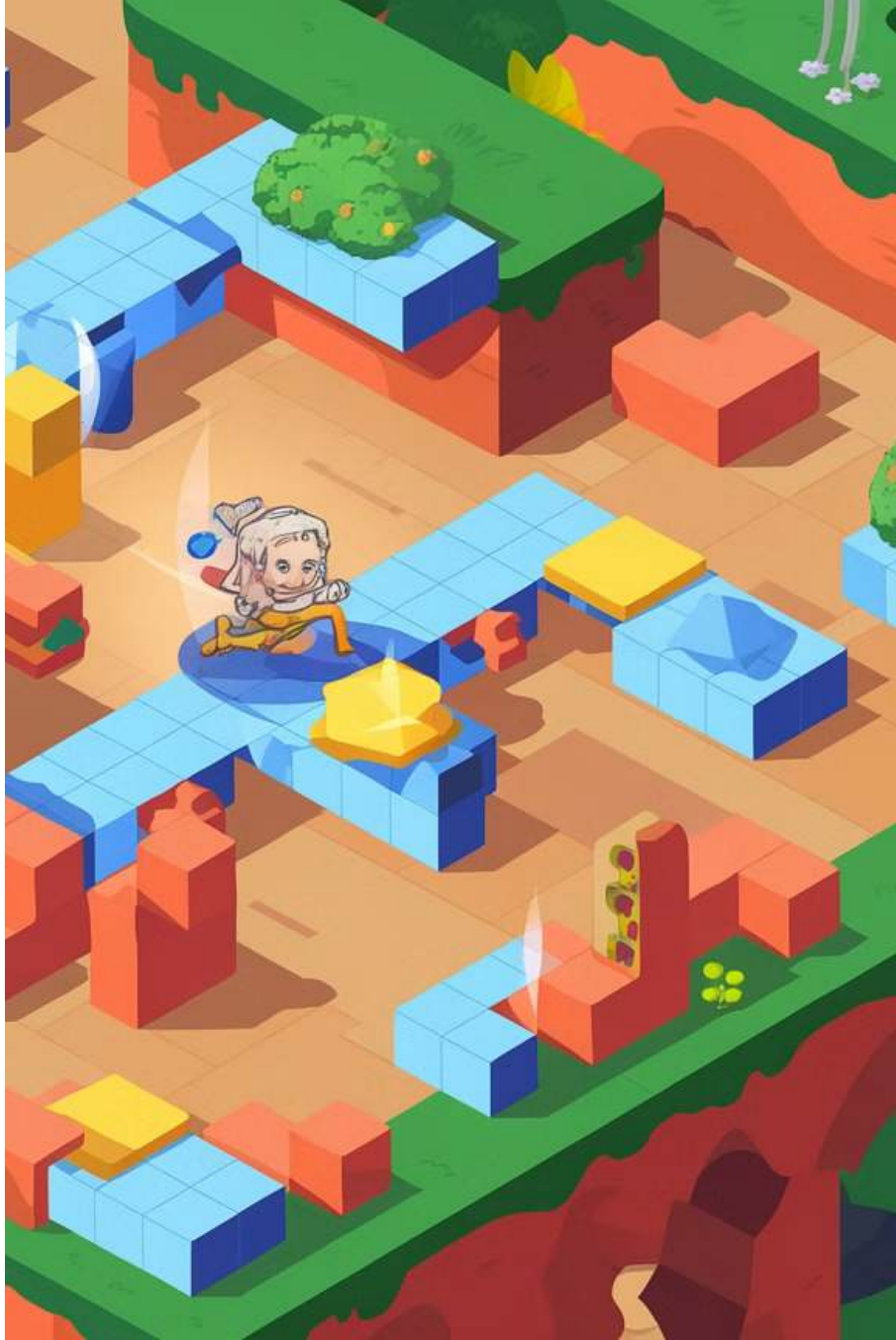
Project	Key Features	Skills Developed
Pong	Ball physics, AI opponent	Core mechanics, simple AI
Puzzle/Action	Level design, enemy AI	Game design, complex AI
Shooter	Player movement, enemy spawning	Input handling, game balance

Pong Game Development

- **Basic Pong:** Implement the classic Pong mechanics, including paddle movement, ball physics, and scoring system.
- **AI Opponent:** Create a smart AI-controlled paddle that can track the ball and hit it back, providing a challenging single-player experience.
- **Power-Ups:** Introduce power-ups like speed boosts, larger paddles, or sticky balls to add more excitement and strategic depth to the game.
- **Multiplayer Mode:** Develop a multiplayer mode where two players can compete against each other, adding a social and competitive element to the Pong experience.

Puzzle/Action Game Development

- **Level Design:** Create intricate levels with obstacles and puzzles to challenge players' problem-solving skills. Incorporate dynamic elements like moving platforms, switches, and hazards to keep gameplay engaging.
- **Game Mechanics:** Incorporate diverse movement mechanics like jumping, climbing, and grappling to allow fluid navigation and interaction with the game world.
- **Enemy AI:** Create strategic AI-controlled enemies that adapt to the player's actions and complement the puzzle-solving gameplay.
- **Scoring and Level Progression:** Incorporate a scoring system that rewards players for solving puzzles efficiently. Link this to a progression system that unlocks new challenges and complexity, creating a sense of accomplishment.



Shooter Game Development

- **Player Movement and Shooting:** Implement smooth, responsive player movement and accurate, satisfying shooting mechanics that feel intuitive and empowering.
- **Enemy Spawning and Movement:** Create dynamic enemy spawning patterns and advanced AI-driven movement behaviors that challenge the player's reflexes and strategic thinking.
- **Collision Detection and Damage:** Implement precise collision detection between bullets, enemies, and the player, ensuring fair and impactful damage application that provides clear feedback.
- **Health and Power-Ups:** Develop a comprehensive health and power-up system that allows players to strategically manage their resources and gain temporary advantages in the heat of combat.

Advanced Game Development Skills

1 Code Refactoring

Improve code structure and readability. Break down complex functions into smaller, manageable parts.

2 Unit Testing

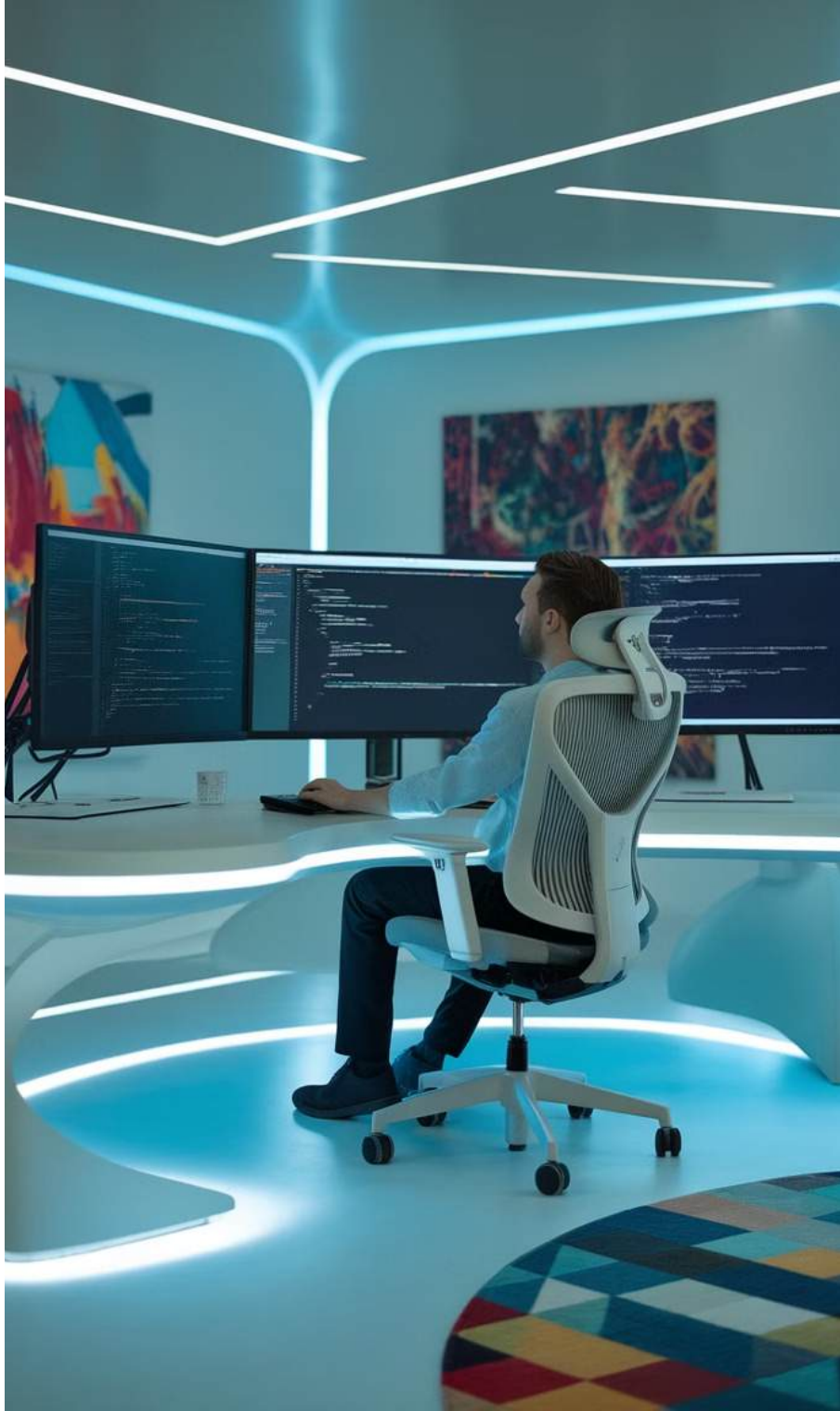
Write tests to verify code functionality. Catch bugs early in the development process.

3 Performance Optimization

Identify and eliminate performance bottlenecks. Create smooth, efficient game experiences.

4 Design Patterns

Learn and apply common game design patterns. Solve recurring problems with proven solutions.



Competencies for Game Development

Programming Skills

Develop proficiency in P5.js to write clean, structured code. Leverage advanced programming techniques like data structures to build robust game systems.

Problem-Solving Skills

Break down intricate challenges into manageable steps. Debug and troubleshoot code effectively to identify and resolve issues.

Creative Thinking

Design engaging game mechanics, experiment with diverse art styles and sound effects, and create immersive gaming experiences.

Collaboration Skills

Work effectively with others to create collaborative projects. Provide and receive constructive feedback to improve the final product.

Measurable Outcomes

1

Complex Games

Develop advanced games with complex features and mechanics.

2

Physics and AI

Implement realistic physics and challenging AI behaviors.

3

Level Design

Create engaging, well-structured game levels with a progression of challenges.

4

Code Refinement

Improve code structure and readability for better maintainability.

5

Unit Testing

Write comprehensive tests to ensure code quality.

6

Collaborative Projects

Work together to create larger-scale game projects.