3D Game Development Microcredential

Overview

The 3D Game Development Microcredential provides a rapid deep dive into Unity development and game design principles. Students will engage with various Unity systems while creating a tech demo showing what they've learned. The course will culminate in a group game development project where like-minded students collaborate to create a working game prototype. Industry collaboration depends on industry partner offerings but may include industry training, meeting professionals in different game dev roles, studio tours, office hours with industry professionals, and feedback on the final game prototypes. Course instruction will run for the first week. Industry insights will run for the majority of the second week. The group collaborative project will begin at the end of the instruction time and be completed over the time culminating in a showcase event. Students must work together to complete the final prototype and will have asynchronous access to the instructors after the first week of instruction.

Learning Outcomes

1. Students should be able to speak about 3D game development and its many aspects competently
2. Students should be able to competently use the Unity 3D interface including functionalities which extend Unity beyond the scope of the course
3. Students should understand the core concepts of the game development process
4. Students should be able to confidently make a Unity game and speak about its design and the several dimensions of game development related to the development of their game including:
    a. Environment Design
    b. Mechanics Design
    c. Lighting and Rendering
    d. Animation and Character Design
    e. Special Effects
    f. Sound and Sound Design
    g. Optimization and Performance

Deliverables

1. Collaborative Pitches (due during the first week)
    a. Students will begin working on pitches for their collaborative projects
2. Instruction Prototype (due at the end of the first week)
    a. This is a self-learning prototype, a simple game tech demo that covers the topics learned during the first week of direct instruction.

    b. Students will work along with the lectures to generate a simple scene that has elements throughout the first week of instruction
    c. Students will have time each day to work on this tech demo to explore concepts from the instruction given that day.
3. Collaborative Game Prototype (due at the end of the micro-credential on August 27th for presentation at the closing event on August 28th)
    a. In the latter half of the first week of instruction, students will pitch ideas to each other and form groups.
    b. Students will be required to work in groups to complete a game prototype
        i. *This means not every student's full ideas may be covered in the final pitch. We encourage students to also work on their ideas after the course!*
        ii. Students will combine ideas into a final pitch and establish group members and roles.
    c. The collaborative game prototype should be a completed playable prototype with interaction and mechanics
    d. This prototype must be presented and demoed by the team members at the closing event

Course

Tentative Microcredential Schedule

August 2026						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
16	17	18	19	20	21	**22**
23	**24**	**25**	**26**	**27**	28	29

Legend			
Instruction	**Industry**	**Group Project Work**	**Closing Event/Showcase**
		Red Font	

Tentative Instruction Schedule

Daily, Monday to Friday - August 17 to August 21

- ☐ 09:00am to 10:30am Lecture slot 1
- ☐ 10:25am to 10:45am Morning Break
- ☐ 10:45am to 12:00pm Lecture slot 2
- ☐ 12:00pm to 01:00pm Lunch break
- ☐ 01:00pm to 02:30pm Lecture slot 3

- [ ] 02:25pm to 02:45pm Afternoon break
- [ ] 02:45pm to 04:30pm Lab

Modules and their order are subject to revision. Module priority is given to modules 1-6 covering fundamentals. Module coverage and depth are dependent on time and progress in the course.

- [ ] Day 1
    - [ ] Module 1: Game Design and Unity Fundamentals
    - [ ] Module 2: Collaboration and Group Work
- [ ] Day 2
    - [ ] Module 3: The Mathematics Behind 3D Games
    - [ ] Module 4: Modelling and Scenes
- [ ] Day 3
    - [ ] Module 5: Animation - Kinematics and Rotations
    - [ ] Module 6: Physical Interactions in Games
- [ ] Day 4
    - [ ] Module 7: Character Control
    - [ ] Module 8: Mechanics, Interaction, and Mechanics
    - [ ] Module 9: High-Level Interaction: From Title Screen to End Game
    - [ ] Module 10: Special Effects
- [ ] Day 5
    - [ ] Module 11: Rendering
    - [ ] Module 12: Sound
    - [ ] Module 13: Performance: Tips & Tricks and Using the Unity Profiler
    - [ ] Group Pitches and Group Selection

Tentative Industry Insights Schedule

Monday to Friday - August 24 to August 27
Blended
Collaborative group project work with mixed asynchronous access to Industry and Instructors
Industry events, tours, training, group project assistance, etc dependent on Industry partner availability

Tentative Closing Event/Showcase

Friday, August 28th
All assignments/materials are due August 27th by 4pm Pacific
Groups will be scheduled to present before the event, participants must demo prototype at event

Module 1: Game Design and Unity Fundamentals

Learning Outcomes

- Students should understand high-level graphics and game concepts
    - The applications of graphics and games
    - The high-level history of graphics, games, and interactive visual technologies
    - Basic graphics and visualization principles such as modelling, sound, rendering, texturing, animation, and procedural generation
    - Basic game design principles
        - The process of design and revisions
        - Developing storyboards
        - Prototyping and Zero-art/Zero-draft demonstration
        - Developing concept art
- Students should understand and be competent with
    - How to setup, create, open and save a project
    - The basic Unity interface
    - The Unity menus
    - The Unity windows and their relationship to the scene and objects

Content

1. Introduction to the Teaching Team
2. Overview of the course and schedule
3. Introduction to graphics, interactive visual technology, and games
    - Concepts
    - History
    - Areas of applications and research
4. Introduction to graphics and visualization concepts
    - Technical concepts
        - Modelling
        - Rendering
        - Animation
        - Interaction
    - Examples of Research
        - Fluid Simulation
        - Character Animation
        - Crowd Simulation
        - Sound Simulation
        - Texture Synthesis
5. Introduction to Game Development
    - Overview of process
    - Zero-draft sketches
    - Storyboarding

    - Game Design Documents
    - Conceptualizing and building a playable demo with zero-graphics and mockups
    - Story writing
    - Iteration and development
    - Basic genres, enumerating common game genres
    - Basic mechanics, enumerating common mechanics
6. Introduction to Unity
    - Creating a new project
    - Overview of the Unity Interface
    - Overview of the Unity Menus
    - Overview of the Unity Windows
    - Basic Scene and GameObject concepts in Unity

Module 2: Collaboration and Group Work

Learning Outcomes

- Students should understand the basic utility and use of Distributed Version Control Systems
- Students should be able to perform basic management of a git repository in a collaborative environment (creating a repo, cloning a repo, pushing to a repo, merging and rebasing)
- Students should understand the reasoning behind the official GitHub Unity .gitignore file and why these files in particular are in the ignore file.

Content

- Distributed Version Control Systems for software management and redundancy
- Basics of git in the Context of Unity Projects
- Using git in a Team

Module 3: The Mathematics Behind Games

Learning Outcomes

- Students should understand the basic building blocks of real-time graphics
    - should be able to identify the use of the CPU and GPU in graphics applications
    - Should understand the reasoning and utility of and for real-time graphics primitives, like points, lines, and triangles
- Students should understand the linear algebra underlying all graphics applications
    - Students should be able to competently speak about and use elementary affine transformations on graphics primitives
- Students should be able to combine transformations and understand the model matrix

- Students should understand from a high level the real-time graphics processing pipeline and what each step achieves

Content

1. Basic overview of matrix math and multiplication
    a. A focus on understanding the utility of representing systems of equations
2. Introduction to elementary affine transformations
3. Introduction to homogeneous coordinates and elementary affine transformations
4. Overview of the real-time graphics processing pipeline from vertices to pixels
5. Overview of what makes 3D games “3D”: the projection matrix, clipping, and the z-buffer algorithm
6. Rasterization, pixels, and colour

Module 4: Modelling and Scenes

Learning Outcomes

- Students should understand how to make basic objects
    - By compositing Unity primitive shapes
    - By making use of the Unity Pro Builder
    - By making models in Blender and importing them
    - Colouring objects via texturing
- Students should understand basic materials concepts and how they bring together colour information
    - Shaders as colouring rules
    - Textures as colouring sources

Content

- Introduction to the Unity Scene Concept
- Introduction to Unity Primitives
- Introduction to Pro Builder and Shape Editing
- Introduction to materials and texturing
- Introduction to Blender

Module 5: Basic Animation - Kinematics and Rotations

Learning Outcomes

- Students should understand the basic principles of animation
    - A system has degrees of freedom, these DoF evolve over time
- Students should understand the classic principles of animation
- Students should be able to keyframe simple animations in Unity

- Students should be able to create and animation basic objects using the unity animation system and animation curves

Content

- Basic Principles of Animation
- Classic Principles of Animation à la Disney
- Keyframing
- Basic Animation Systems in Unity
    - Animation Controller
    - Animation Clips in the Animation Window
    - Animation States in the Animator Window
- Key Framing in Unity
- Mecanim and Human Animation
    - Basic Principles of Humanoid Animation
    - Motion Capture
    - Retargeting

Module 6: Physical Interactions in Games

Learning Outcomes

- Students should be able to effectively use PhysicsMaterials to control friction and bounce interactions at contact points
- Students should be able to capture and respond to collision events in code

Content

- The basic mathematics behind second-order simulation systems
    - Recapping basic physics
    - Moving objects with physics
- Basic Collision concepts
    - Detection
    - Resolution
- Introduction to the Unity Physics System
    - Collider Shapes
    - Separating Graphics and Physics in the Scene Hierarchy, Best Practices

Module 7: Character Control

Learning Outcomes

- Students should be able to discuss the basics of character control

- Students should be able to capture keyboard and mouse input and translate user intentions in the character movement in a 3D environment
- Student should be able to implement simple 3D platformer movement, using classic “tank” style controls

Content

1. Introduction to the Player Character and Controller Types
    a. Handling Player input
    b. Using RigidBody physics to move characters in the simulation loop
    c. PhysicsMaterials and issues with contact simulation
2. Introduction to the Non-Player Character (NPC)
    a. Basic NPC AI concepts
        i. State machines
        ii. Behaviour trees
    b. Moving an NPC using a state machine
3. Tuning character motion and interaction

Module 8: Mechanics, Interaction, and Scoring Mechanisms

Learning Outcomes

- Students should be able to identify and discuss basic game mechanics in game design
- Students should be able to code simple keyboard-based character functionalities
- Students should be able to detect physical interactions between objects and between their character and the world
- Students should be able to combine prior knowledge about animation systems with new information regarding scripting interactions to make their world interactive
- Students should be able to accumulate score information and show it in the debug system

Content

- Adding Functionality to Your Character
- Coding Basic Interactions
- Pickups and Switches: Using the Unity Physics System to Capture Interactions
- Keeping Track of Data: Scoring Mechanisms

Module 9: High-Level Interaction: From Title Screen to End Game

Learning Outcomes

- Students should be able to interface with basic application-level code
- Students should understand the Unity User Interface canvas system

- Students should be able to create simple User Interfaces
- Students should be able to load scenes and exit their game application from a menu
- Students should understand the complexity of saving game state and should be able to work with basic PlayerPrefs to save simple game settings

Content

1. App-level code
    a. Scripting basic exit keys for testing
2. Introduction to Unity UI and Canvas System
3. Developing a basic menu screen
4. Scene loading to move between menus, screens, and loading times
5. Triggering end-game logic
6. Basic use of PlayerPrefs for Quality settings and Volume
7. Saving and loading games

Module 10: Special Effects/VFX

Learning Outcomes

- Students should understand the utility of special effects in compositing scenes, driving narrative ideas, and layering interactivity
-

Content

1. Introduction to Particle Systems
2. Introduction to Post-Processing effects
3. Using particle systems globally and on triggered events to layer depth into a game
4. Tuning post-processing effects and triggering effects

Module 11: Rendering

Learning Outcomes

- Students should be able to discuss basic lighting concepts and the underlying conceptual rendering frameworks that govern the look of most games
- Students should be able to install and modifying the postprocessing effects stack to achieve different Look-and-feel

Content

1. Introduction to Rendering concepts
    a. Surfaces

    b. Light transport
    c. The rendering equation
    d. Different lighting paradigms, i.e. solutions to the rendering equation
        i. Lighting in the real-time pipeline
            1. Forward rendering
            2. Deferred Rendering
        ii. Introduction to Ray tracing
        iii. Introduction to Ray marching
2. Introduction to Unity's lighting system
    a. Adding different types of lights
    b. Baking lighting on static objects
3. Introduction to Unity's new Universal Render Pipeline
4. Diving into materials and shaders

Module 12: Sound

Learning Outcomes

- Students will be familiar with the basic principles of digital audio
- Students should understand the Unity audio systems and how to add them to the world
- Students should understand how to trigger and instantiate audio from code
- Students should understand the utility of and be able to apply fall-off functions to sounds
- Students should be familiar with the Unity Audio Mixer and the process of mixing audio and applying audio effects

Content

1. Introduction to Unity's sound system
    - Audio sources and the audio listener
2. Editing Audio sources
    - Fall off curves
    - The Audio Mixer
        - Mixing
        - Filter Effects
3. Background audio versus triggered audio

Module 13: Performance: Tips & Tricks and Using the Unity Profiler

Learning Outcomes

- Students should understand common performance pitfalls in Unity and C# development concerning real-time game development

- Students should be able to competently solve performance issues and be able to find insight from the Unity Docs and training on best practices
- Students should be able to use the Unity Profiler to investigate performance issues
- Students should be able to identify CPU vs GPU-bound games and their underlying issues

Content

1. Overview of performance issues
    a. Breaking performance issues into CPU and GPU bottlenecks
    b. Best practices in Unity and C# development
    c. Handling garbage collection issues via best practice
2. Pre-instantiating objects at load time, repeated object Instantiation, and object pooling
3. Mesh Level or Detail
    a. Modifying an existing mesh to create multiple levels of detail versions
    b. Setting up LOD models in Unity's built-in LOD system
4. Modifying asset import and compression settings for performance and packaging
5. The Unity Profiler as a key tool for investigating and addressing performance issues